CODING AND ROBOTICS

National Curriculum Statement (NCS)

Curriculum and Assessment Policy Statement

CAPS
STRUCTURED. CLEAR. PRACTICAL
HELPING TEACHERS UNLOCK THE POWER OF NCS

INTERMEDIATE PHASE
GRADE 4 – 6

basic education
Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

Read to Lead
A Reading Nation is a Leading Nation

2030 NDP

30 YEARS OF FREEDOM
South Africa 1994 - 2024

Department of Basic Education

222 Struben Street

Private Bag X895

Pretoria 0001

South Africa

Tel: +27 12 357 3000

Fax: +27 12 323 0601

120 Plein Street

Private Bag X9023

Cape Town 8000

South Africa

Tel: +27 21 465 1701

Fax: +27 21 461 8110

Website: http://www.education.gov.za

© 2024 Department of Basic Education

ISBN: 978-1-4315-3989-5

Design and Layout by: Department of Basic Education

CURRICULUM AND ASSESSMENT POLICY STATEMENT (CAPS)

CODING AND ROBOTICS

INTERMEDIATE PHASE
GRADE 4 – 6

FOREWORD BY THE MINISTER

Our national curriculum is the culmination of our efforts over a period of seventeen years to transform the curriculum bequeathed to us by apartheid. From the start of democracy, we have built our curriculum on the values that inspired our Constitution (Act 108 of 1996). The Preamble to the Constitution states that the aims of the Constitution are to:

- *heal the divisions of the past and establish a society based on democratic*
- *values, social justice and fundamental human rights;*
- *improve the quality of life of all citizens and free the potential of each person;*
- *lay the foundations for a democratic and open society in which government is based on the will of the people and every citizen is equally protected by law; and*
- *build a united and democratic South Africa able to take its rightful place as a sovereign state in the family of nations.*

Education and the curriculum have an important role to play in realising these aims. In 1997 we introduced outcomes-based education to overcome the curricular divisions of the past, but the experience of implementation prompted a review in 2000. This led to the first curriculum revision: the Revised National Curriculum Statement Grades R-9 and the National Curriculum Statement Grades 10-12 (2002).

Ongoing implementation challenges resulted in another review in 2009 and we revised the Revised National Curriculum Statement (2002) and the National Curriculum Statement Grades 10-12 to produce this document.

From 2012 the two National Curriculum Statements, for Grades R-9 and Grades 10-12 respectively, are combined in a single document and will simply be known as the National Curriculum Statement Grades R-12. The National Curriculum Statement for Grades R-12 builds on the previous curriculum but also updates it and aims to provide clearer specification of what is to be taught and learnt on a term-by-term basis.

The National Curriculum Statement Grades R-12 represents a policy statement for learning and teaching in South African schools and comprises of the following:

(a) Curriculum and Assessment Policy Statements (CAPS) for all approved subjects listed in this document;

(b) National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R-12; and

(c) National Protocol for Assessment Grades R-12.

MRS ANGIE MOTSHEKGA, MP

MINISTER OF BASIC EDUCATION

CONTENTS

TABLES AND FIGURES

Tables

Figures

1 SECTION 1
INTRODUCTION TO THE CURRICULUM AND ASSESSMENT POLICY STATEMENT FOR CODING AND ROBOTICS INTERMEDIATE PHASE (GRADE 4 – 6)

1.1 BACKGROUND

The *National Curriculum Statement Grades R – 12 (NCS)* stipulates policy on curriculum and assessment in the schooling sector.

To improve implementation, the National Curriculum Statement was amended, with the amendments coming into effect in January 2012. A single comprehensive Curriculum and Assessment Policy document was developed for each subject to replace Subject Statements, Learning Programme Guidelines and Subject Assessment Guidelines in Grades R - 12.

1.2 OVERVIEW

(a) The *National Curriculum Statement Grades R – 12 (January 2012)* represents a policy statement for learning and teaching in South African schools and comprises the following:

- (i) National Curriculum and Assessment Policy Statements for each approved school subject*;*
- (ii) The policy document, National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12; and
- (iii) The policy document, National Protocol for Assessment Grades R – 12 (January 2012).

(b) The *National Curriculum Statement Grades R – 12 (January 2012)* replaces the two current national curricula statements, namely the

- *(i)* *Revised National Curriculum Statement Grades R - 9, Government Gazette No. 23406* of 31 May 2002, and
- (ii) *National Curriculum Statement Grades 10 - 12 Government Gazettes, No. 25545 of 6 October 2003* and *No. 27594* of 17 May 2005.

(c) The national curriculum statements contemplated in subparagraphs (a) and (b) comprise the following policy documents which will be incrementally repealed by the *National Curriculum Statement Grades R – 12 (January 2012)* during the period 2012-2014:

- (i) The Learning Area/Subject Statements, Learning Programme Guidelines and Subject Assessment Guidelines for Grades R - 9 and Grades 10 – 12;
- (ii) The policy document, *National Policy on assessment and qualifications for schools in the General Education and Training Band d*, promulgated in *Government Notice No. 124* in *Government Gazette No. 29626* of 12 February 2007;
- (iii) The policy document, the *National Senior Certificate: A qualification at Level 4 on the National Qualifications Framework (NQF)*, promulgated in *Government Gazette No.27819* of 20 July 2005;
- (iv) The policy document, *An addendum to the policy document, the National Senior Certificate: A qualification at Level 4 on the National Qualifications Framework (NQF), regarding learners with special needs*, published in *Government Gazette, No.29466* of 11 December 2006, is incorporated in the policy document, *National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12; and*
- (v) The policy document, *An addendum to the policy document, the National Senior Certificate: A qualification at Level 4 on the National Qualifications Framework (NQF), regarding the National Protocol for Assessment (Grades R – 12)*, promulgated in *Government Notice No.1267* in *Government Gazette No. 29467* of 11 December 2006.

(c) The policy document, *National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12*, and the sections on the Curriculum and Assessment Policy as contemplated in Chapters 2, 3 and 4 of this document, constitute the norms and standards of the *National Curriculum Statement Grades R – 12.* It will therefore, in terms of *section 6A* of the *South African*

Schools Act, 1996 (Act No. 84 of 1996,) form the basis for the Minister of Basic Education to determine minimum outcomes and standards, as well as the processes and procedures for the assessment of learner achievement to be applicable to public and independent schools.

1.3 General aims of the South African Curriculum

- The *National Curriculum Statement Grades R - 12* gives expression to the knowledge, skills and values worth learning in South African schools. This curriculum aims to ensure that children acquire and apply knowledge and skills in ways that are meaningful to their own lives. In this regard, the curriculum promotes knowledge in local contexts, while being sensitive to global imperatives.
- The National Curriculum Statement Grades R - 12 serves the purposes of:
  - equipping learners, irrespective of their socio-economic background, race, gender, physical ability or intellectual ability, with the knowledge, skills and values necessary for self-fulfilment, and meaningful participation in society as citizens of a free country;
  - providing access to higher education;
  - facilitating the transition of learners from education institutions to the workplace; and
  - providing employers with a sufficient profile of a learner's competences.
- The National Curriculum Statement Grades R - 12 is based on the following principles:
  - Social transformation: ensuring that the educational imbalances of the past are redressed, and that equal educational opportunities are provided for all sections of the population;
  - Active and critical learning: encouraging an active and critical approach to learning, rather than rote and uncritical learning of given truths;
  - High knowledge and high skills: the minimum standards of knowledge and skills to be achieved at each grade are specified and set high, achievable standards in all subjects;
  - Progression: content and context of each grade shows progression from simple to complex;
  - Human rights, inclusivity, environmental and social justice: infusing the principles and practices of social and environmental justice and human rights as defined in the Constitution of the Republic of South Africa. The National Curriculum Statement Grades R – 12 is sensitive to issues of diversity such as poverty, inequality, race, gender, language, age, disability and other factors;
  - Valuing indigenous knowledge systems: acknowledging the rich history and heritage of this country as important contributors to nurturing the values contained in the Constitution; and
  - Credibility, quality and efficiency: providing an education that is comparable in quality, breadth and depth to those of other countries.
- The National Curriculum Statement Grades R - 12 aims to produce learners that can:
  - identify and solve problems and make decisions using critical and creative thinking;
  - work effectively as individuals and with others as members of a team;
  - organise and manage themselves and their activities responsibly and effectively;
  - collect, analyse, organise and critically evaluate information;
  - communicate effectively using visual, symbolic and/or language skills in various modes;
  - use science and technology effectively and critically showing responsibility towards the environment and the health of others; and
  - demonstrate an understanding of the world as a set of related systems by recognising that problem solving contexts do not exist in isolation.
- Inclusivity should become a central part of the organisation, planning and teaching at each school. This can only happen if all teachers have a sound understanding of how to recognise and address barriers to learning, and how to plan for diversity.

The key to managing inclusivity is ensuring that barriers are identified and addressed by all the relevant support structures within the school community, including teachers, District-Based Support Teams, Institutional-Level Support Teams, parents and Special Schools as Resource Centres. To address barriers in the classroom, teachers should use various curriculum differentiation strategies such as those included in the Department of Basic Education's *Guidelines for Inclusive Teaching and Learning* (2010).

1.4 Time Allocation

1.4.1 Foundation Phase

(a) The instructional time in the Foundation Phase is as follows:

Subject	Grade R (Hours)	Grades 1-2 (Hours)	Grade 3 (Hours)
Home Language	10	7/8	7/8
First Additional Language		2/3	3/4
Mathematics	7	7	7
Life Skills • Beginning Knowledge • Creative Arts • Physical Education • Personal and Social Well-being	**5** (1) (1,5) (1,5) (1)	**5** (1) (1,5) (1,5) (1)	**5** (1,5) (1,5) (1) (1)
Coding and Robotics	(1)	(1)	(2)
Total	**23**	**23**	**25**

(b) Instructional time for Grades R, 1 and 2 is 23 hours and for Grade 3 is 25 hours.

(c) Ten hours are allocated for languages in Grades R-2 and 11 hours in Grade 3. A maximum of 8 hours and a minimum of 7 hours are allocated for Home Language and a minimum of 2 hours and a maximum of 3 hours for Additional Language in Grades R – 2. In Grade 3 a maximum of 8 hours and a minimum of 7 hours are allocated for Home Language and a minimum of 3 hours and a maximum of 4 hours for First Additional Language.

(d) In Life Skills Beginning Knowledge is allocated 1 hour in Grades R – 2 and 2 hours as indicated by the hours in brackets for Grade 3.

1.4.2 Intermediate Phase

The instructional time in the Intermediate Phase is as follows:

Subject	Hours
Home Language	6
First Additional Language	5
Mathematics	6
Natural Sciences	2,5
Social Sciences	3
Life Skills • Creative Arts • Physical Education • Personal and Social Well-being	**3** (1) (1) (1)
Coding and Robotics	2
Total	**27,5**

1.4.3 Senior Phase

(a) The instructional time in the Senior Phase is as follows:

Subject Choice: Option 1	Subject Choice: Option 2	Hours
Home Language	Home Language	5
First Additional Language	First Additional Language	4
Mathematics	Mathematics	4,5
Natural Science	Natural Science	3
Social Sciences	Social Sciences	3
*Technology	*Economic Management Sciences	2
Coding and Robotics	Coding and Robotics	2
Life Orientation	Life Orientation	2
Creative Arts	Creative Arts	2
Total		**27,5**

* Schools/Learners can follow Option 1 (MST Stream) or Option 2 (Business Stream)

1.4.4 Grades 10-12

(a) The instructional time in Grades 10-12 is as follows:

Subject	Time allocation per week (hours)
I. Home Language	**4.5**
II. First Additional Language	**4.5**
III. Mathematics	**4.5**
IV. Life Orientation	**2**
V. A minimum of any three subjects selected from Group B Annexure B, Tables B1-B8 of the policy document, *National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12,* subject to the provisos stipulated in paragraph 28 of the said policy document.	**12 (3x4h)**

The allocated time per week may be utilised only for the minimum required NCS subjects as specified above and may not be used for any additional subjects added to the list of minimum subjects. Should a learner wish to offer additional subjects, additional time must be allocated for the offering of these subjects.

2 SECTION 2: DEFINITION, AIMS, SKILLS AND CONTENT

2.1 INTRODUCTION

Coding and Robotics represents an interdisciplinary and multidisciplinary subject that integrates various components of STEAM (Science (including Computer Science), Technology, Engineering, Arts, and Mathematics).

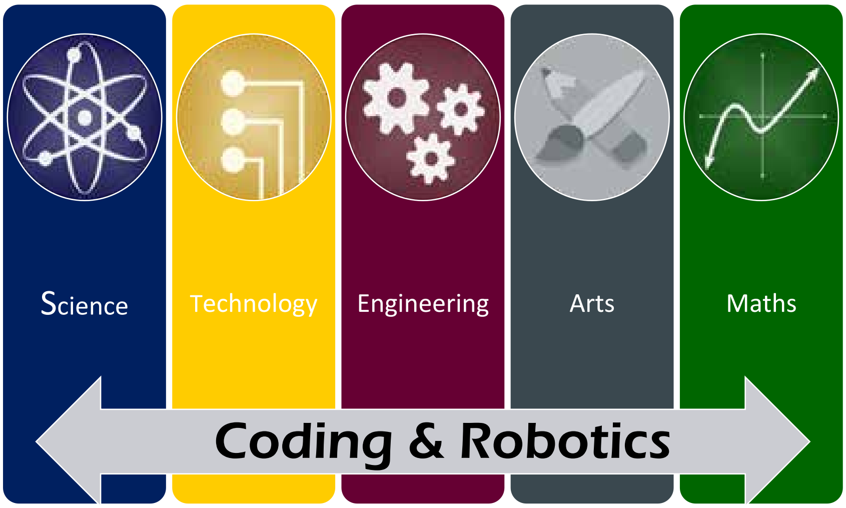


Figure 2.1 Coding and Robotics as a STEAM discipline

The main driving force behind the uptake and surge of Coding and Robotics as a subject at school level is the link to the 4^{th} and 5^{th} industrial revolution (4IR, and 5IR). In the context of this curriculum the focus resides in the grounding concepts of STEAM related subjects.

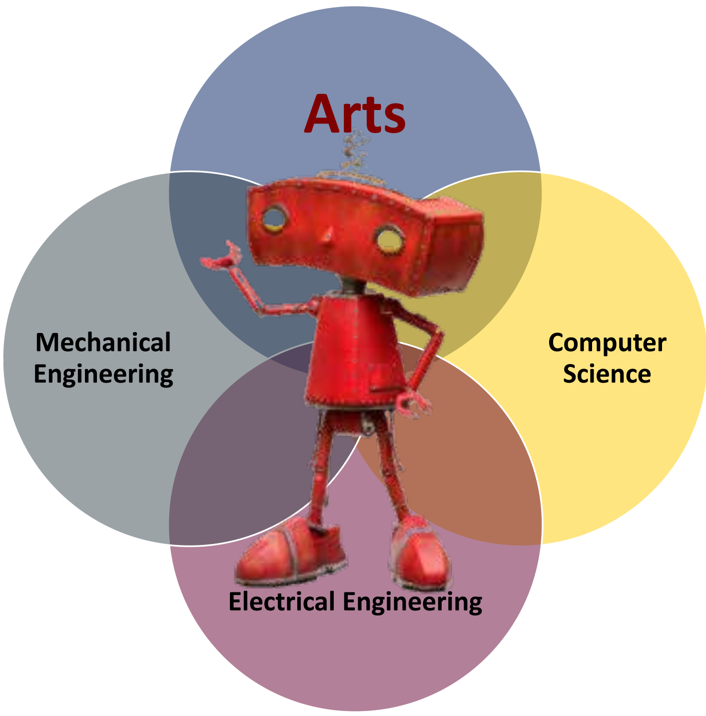


Figure 2.2: Coding and Robotics as a multi-disciplinary subject

2.2 WHAT IS CODING AND ROBOTICS

Coding and Robotics combine the principles of programming with the design, construction, and operation of robots. Programming concepts, practices, and perspectives are applied to control devices to perform specific tasks. It includes digital concepts that refer to various ideas, principles and processes that are associated with digital technologies and their use.

The Coding and Robotics curriculum is based on the following pillars as depicted in the figure below.

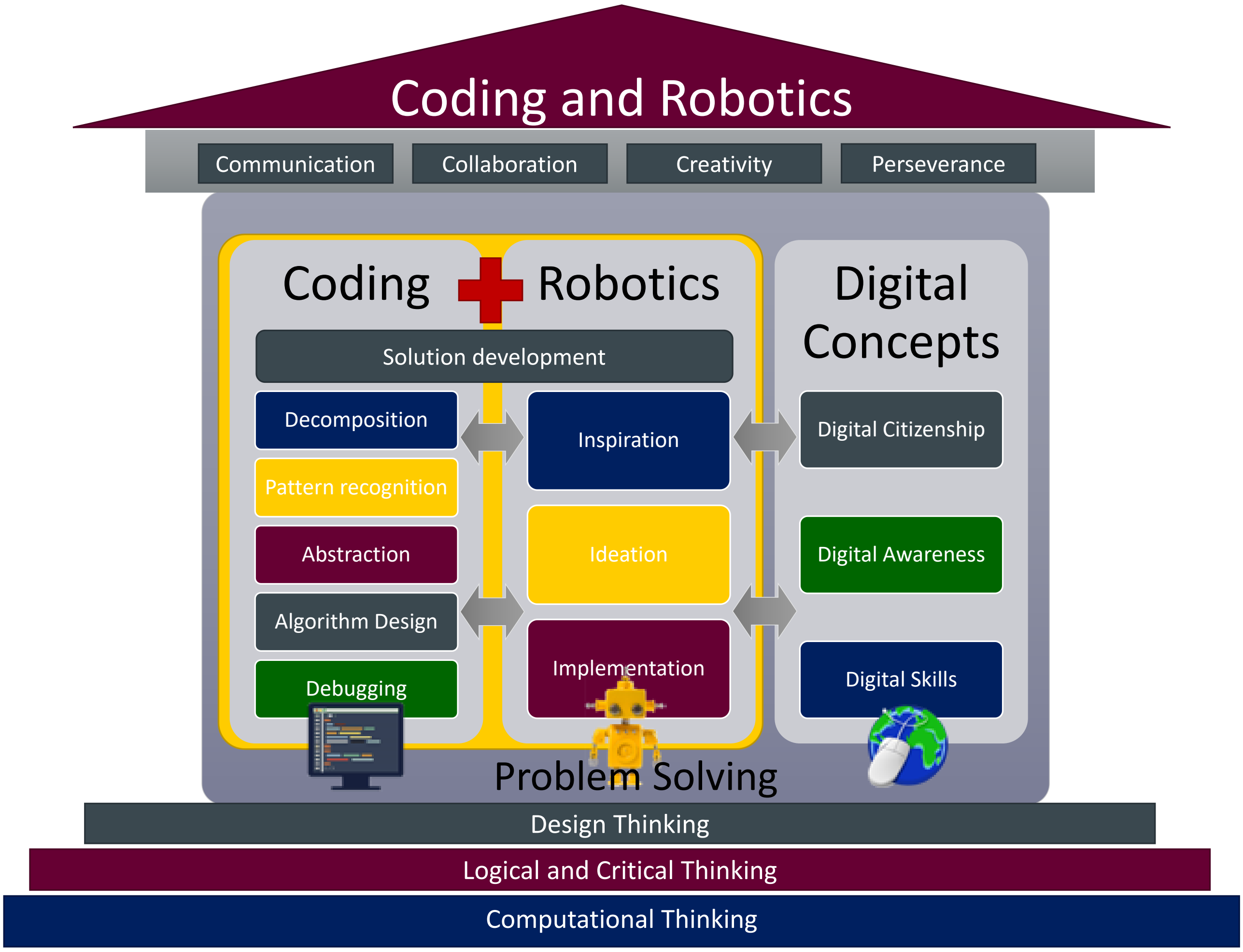


Figure 2.3: Overview of Coding and Robotics as a Subject

Coding is the process of creating a logical set of instructions that a human or a computing device can understand and execute, which require a deep understanding of computational thinking and problem solving.

Robotics deals with the design, operation, and use of devices and robots that can be programmed to perform tasks autonomously or semi-autonomously or by direct control. It presents the learners with the opportunity to see their thinking, design, and code in action.

Digital concepts encompass a range of digital literacy skills and awareness that enables learners to leverage digital technologies to their fullest potential and use digital tools responsibly.

2.3 SPECIFIC AIMS

The teaching and learning of Coding and Robotics (C&R) aim to develop the following for the learner to be able to:

- develop computational thinking skills to solve problems.
- advance design thinking to develop creative and human-centred approaches to solve problems.

- become part of a generation of creative, innovative systems thinkers that can use coding, robotics, and digital competencies to express their ideas.
- foster creativity, critical thinking, collaboration, communication, and innovation.
- function ethically and effectively in a digital and information-driven world.
- develop a critical awareness of how technologies impact society at large.
- instil self-efficacy and confidence to deal with situations requiring computational thinking, design thinking and problem solving.
- prepare for future careers in STEAM related fields.
- adopt a culture of being self-directed, life-long learners who can apply their skills in a wide range of contexts and situations (adaptable, flexible and resilient).

2.4 SPECIFIC SKILLS

The following skills are specifically emphasised:

2.4.1 Computational Thinking

Computational thinking is an attitude and a skill set where one uses specific techniques and strategies to complete tasks successfully and to solve problems systematically. It further helps one in arriving at a solution that both humans and a computer can understand.

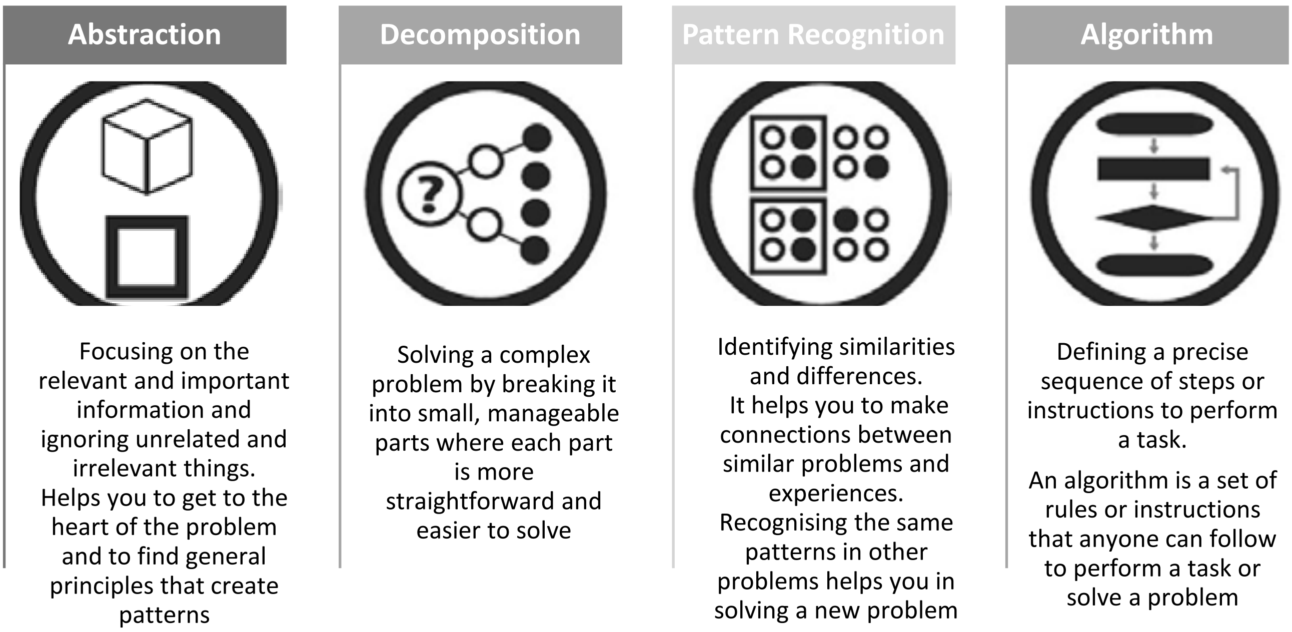


Figure 2.4: Computational Thinking Pillars

In Coding and Robotics, computational thinking helps learners to develop problem-solving strategies which they can apply when developing coding solutions (algorithms) as well as robotics solutions. It can also be applied to solve everyday life.

In terms of robotics, learners are demonstrating computational thinking concepts and practices when designing, constructing, and programming a robot. The robot's performance demonstrates the result of the learner's computational thinking practices as they iteratively test and debug their coding.

2.4.2 Design Thinking

In education, design thinking (DT) refers to a human-centred approach that encourages creativity and innovation when generating user-focused products, services, or experiences. Design Thinking is often expressed as an activity that involves the three **I** processes, namely:

- **Inspiration:** where creative thinking is applied to tackle a problem or challenge at hand, by gaining a deeper understanding of the problem and its context as well as to identify opportunities for innovation.
- **Ideation:** involves the generation of a wide range of ideas and potential solutions using various approaches such as brainstorming, prototyping and experimentation.
- **Implementation:** where the ideas and potential solutions are put into action. It includes testing, getting feedback and subsequent improvements of the design or solution.

Related to the three **Is**, is the notion that Design Thinking is also a problem-solving approach that combines creativity with structure and human-centred methods to understand and tackle challenges which involves empathizing with users, defining their needs, ideating possible solutions, prototyping, and testing those solutions, and iterating based on feedback. The following describes the design process:

- **Empathise:** involves gaining an understanding of who the end user is in a specific context, and how the envisaged solution will be appropriate towards addressing the problem.
- **Define:** relates to specifying in detail what the users' needs are, which could include the goals, skills available, and core principles that will guide the work to be done.
- **Ideate:** pertains to the creation of ideas and solutions using techniques such as brainstorming.
- **Prototype:** concerns the creation of one or several solutions to address the problem at hand.
- **Test:** relates to the process of determining how well the solution solves or address the problem. In this phase, feedback is important as the feedback could be used towards the improvement and enhancement and/or redesign of the complete solution or artefact.

Figure 2.6 depicts the relationship between the Design Thinking and Design Problem Solving approach.

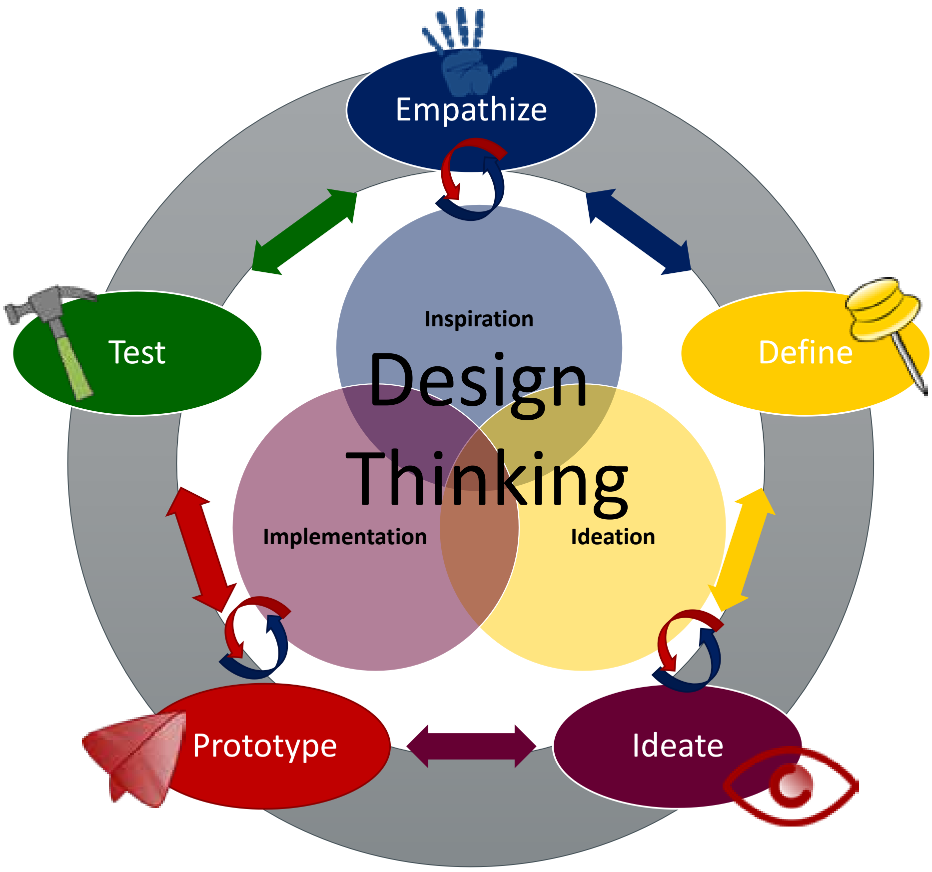


Figure 2 5: Design Thinking and Problem-Solving Process

2.5 HIGH-LEVEL COMPETENCIES – CODING AND ROBOTICS

The three main topical areas of Coding and Robotics each comprises a set of key learning competencies central to their area of focus.

The following diagram outlines the three main topical areas and the main learning competencies, associated at the final stage of curriculum cognition wherein the learner demonstrates competence and proficiency at the appropriate level.

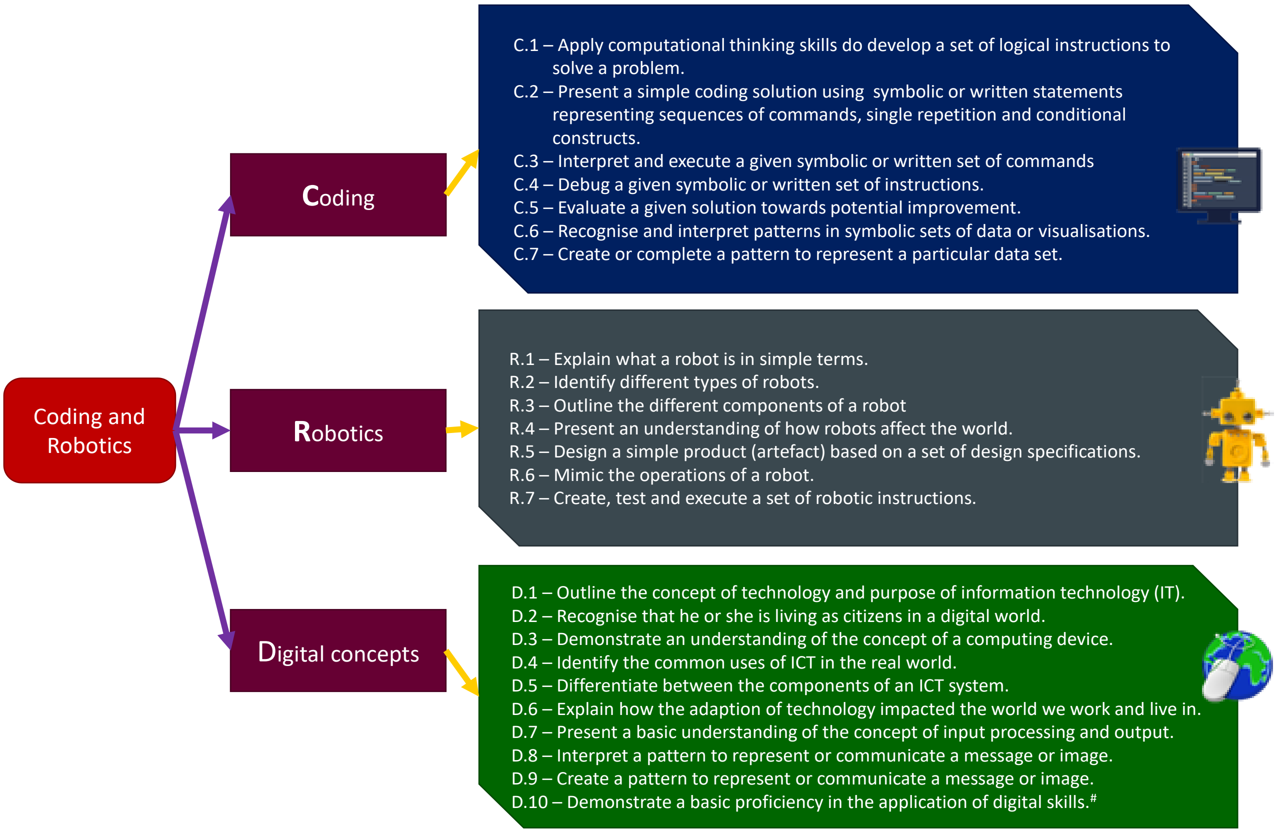


Figure 2 6: High-level Curriculum Competencies

A competence is a combination of knowledge skills, attitudes, and values which is reflected in behaviour that can be observed, measured, and evaluated. It refers to the ability to perform a specific task successfully and efficiently or in a manner that yields desirable outcomes.

2.6 CODING AND ROBOTICS CONCEPTS, PRACTICES AND PERSPECTIVES

2.6.1 Coding

In coding, the following concepts, practices, and perspectives must be developed and practised repeatedly:

Concepts	Practices	Perspectives
• Algorithm • Sequence • Loop (Iteration) • Conditional (Decisions) • Operator • Logic • Data • Event • Debug • Representation • Parallelism • Automation	• Abstraction • Decomposition • Pattern Recognision • Generalisation • Algorithm Design • Incremental Development • Testing and Debugging • Evaluation • Modularise • Logical thinking • Creating computational artefacts	• Expressing and Creating • Questioning • Connecting • Collaboration • Perserverance • Choice of Conduct

Figure 2.7 Coding Concepts, Practices and Perspectives

The table below describes the coding content to be covered and the skills to be developed. Table 2-1 must be read in conjunction with Table 2-4 in Section 2.12.1 and Table 2-7 in Section 2.15.1 for progression per grade.

Table 2-1 Coding content and skills

Concept	Content/Skills
Algorithm	• Definition and importance of algorithms in computer programming. • Characteristics of a good algorithm. • Examples of algorithms in everyday tasks and programming. • Algorithm development using computational thinking.
Sequence	• Understanding the concept of sequential execution of instructions. • Introduction to basic programming constructs like statements and expressions. • Writing simple programs to perform sequential tasks.
Loop (iteration)	• Explanation of loops and their purpose in programming. • Different types of loops (e.g., while loop, for loop) and their syntax. • Examples demonstrating the use of loops for repetitive tasks. • Writing simple programs to perform tasks that include repetition
Conditional (Decision)	• Understanding conditional statements (if…then, if…then…else)) and their role in decision-making. • Using comparison operators in conditional statements. • Writing programs with conditional logic to handle different scenarios.
Operator	• Assignment, comparison, and logical operators. • Precedence and associativity rules for operators. • Use of operators in expressions and assignments in programs
Logic	• Introduction to Boolean logic and truth tables. • Understanding logical operators (AND, OR, NOT) and their application in programming. • Writing programs that implement logical operations and evaluate conditions.
Data	• Types of data (e.g. numbers, string, Boolean) and their use in programming. • Variables and data types. • Input/output and processing operations for data manipulation.
Event	• Concept of events and event-driven programming. • Handling user interactions and system events in programs. • Implementing event handlers in programs.
Debug	• Techniques for identifying and fixing errors in code (debugging). • Using debugging tools and techniques (e.g., trace tables). • Debug common programming errors (syntax errors, logic errors).
Representation	• Understanding data representations (binary). • Exploring the concept of abstraction in programming.
Automation	• Exploring automation concepts and their significance. • Writing scripts to automate repetitive tasks.
Parallelism	• Concept of parallelism using e.g. two scripts, running concurrently, allowing different actions to happen simultaneously (e.g. broadcast & receiving, clones, parallel blocks, e.g. "forever" block can run continuously while other code executes concurrently.

2.6.2 Robotics

In addition to the coding concepts, practices and perspectives, in robotics, the following concepts, practices, and perspectives must be developed and practised repeatedly:

Concepts	Practices	Perspectives
• Motion • Sensor • Actuator • Controller • Logic • Power Source • Automation • Instruction • Communication • Coding (Programming)	• Computational Thinking • Design Thinking • Prototyping • Design and Construction • Algorithm Design • Testing and Reconfiguration • Reflection and Iteration • Creative Thinking • Logical thinking • Creating robotics artefacts	• Expressing and Creating • Innovation • Questioning • Connecting • Collaboration • Perserverance • Choice of Conduct

Figure 2.8 Robotics Concepts, Practices and Perspectives

The following table describes the robotics content to be covered and the skills to be developed. Table 2-2 must be read in conjunction with Table 2-5 in Section 2.12.2 and Table 2-8 in Section 2.15.2 for competencies and progression per grade

Table 2-2 Robotics content and skills

Concept	Content/Skills
Motion	• Introduction to different types of robot motion: linear, rotational, and combined. • Exploring methods of locomotion such as wheels, tracks, legs, and aerial mechanisms.
Sensor	• Overview of sensors used in robotics, including proximity sensors, cameras, ultrasonic, microphone, temperature sensors. • Explanation of sensor principles and how they gather data from the environment. • Applications of sensors in navigation, obstacle avoidance, object detection, and environmental monitoring.
Actuator	• Introduction to actuators responsible for converting electrical energy into mechanical motion. • Types of actuators: electric motors (DC motors, servo motors). • Understanding the role of actuators in robot manipulation, locomotion, and control.
Controller	• Components as part of a robot responsible for controlling the robot, gathering input, and providing output. (Examples: Arduino, Raspberry Pi, Micro: bit)
Logic	• Introduction to logical operations and decision-making in robotics. • Understanding Boolean logic and its application in robot control. • Implementing logical operations for conditional behaviour, state transitions, and autonomous decision-making.
Power Source	• Overview of power sources for robotics, including batteries, and external power supplies (e.g. solar). • Understanding power requirements and considerations for selecting appropriate power sources. • Designing power distribution systems and managing power consumption for optimal robot performance.
Automation	• Explanation of automation in robotics as the process of performing tasks with minimal human intervention. • Applications of automation in manufacturing, logistics, agriculture, healthcare, and service industries. • Designing automated systems using robots for repetitive, dangerous, or labour-intensive tasks.
Instruction	• Understanding instructions as commands given to robots to perform specific actions. • Types of instructions: sequential instructions, conditional instructions, repetitive instructions (loops). • Writing clear and precise instructions for programming robots to accomplish desired tasks.
Communication	• Basic overview of communication technologies used between two or more devices, e.g. Wi-Fi, Bluetooth.
Coding	• Introduction to a robotics programming environment. • Basics of robot programming: variables, data types, control structures (sequence, if statements, loops), functions, and libraries. • Hands-on coding exercises and projects to develop skills in algorithm development and in robot programming.

2.6.3 Digital Concepts

Digital concepts are fundamental ideas and principles that underpin and support coding and robotics. They encompass various aspects of technology and computer science, providing the context and application for these fields. In Coding and Robotics, digital concepts are divided into the following topics: Digital Citizenship, Digital Awareness and Digital Skills. The following must be read in conjunction with Table 2-6 in Section 2.12.3.

2.6.3.1 Digital Citizenship

The rights, responsibilities and behaviours (respect, integrity, and safety) displayed by individuals in the digital world. It encompasses a spectrum of behaviours, spanning from respecting the privacy of others to protecting personal data, being mindful of online threats and ensuring one's safety in the digital sphere.

Responsible behaviour

- Implications of digital citizenship
- Responsible and ethical behaviour in the digital realm.
- Cybersecurity awareness such as protecting personal information online, recognising cyber threads and practising safe online behaviour.
- Privacy & security (strong passwords, not sharing personal information), cyberbullying, digital footprint and netiquette.
- Digital health and welfare.

Information Assessment

- Credibility and reliability of online sources and information.
- Fake news
- Intellectual property and its implications.

Impact and Responsibility

- Awareness of how technology adaptation influence our work and lifestyle
- Consequences and implications of online actions.
- Lasting impact of online content (digital footprint)
- Ethical use of computers, including software and robotics applications.
- Dangers of the online world, computer/cyber crime

Figure 2-9 Digital Citizenship Concepts

Digital Citizenship helps to develop an awareness of responsible and ethical behaviour in the digital world as it provides principles for shaping the digital landscape and influencing individual and collective behaviour online. In Coding and Robotics, understanding digital citizenship is important when developing software and robotics applications to ensure they are used in a responsible and ethical manner.

2.6.3.2 Digital Awareness

The recognition of the competencies, expertise, and the mindset needed by individuals effectively to use digital tools, entail understanding and the applications of technologies in a world that is becoming more interconnected. This emphasises the essential sense of familiarity, adaptability, and proficiency needed for utilising fundamental technology.

Data and Information

- Data collection, storage and processing
- Transformation of data into information.
- Utilisation of data and information for decision-making and innovation.

Computing devices

- Technology vs Information Technology (IT)
- Basic model of a computing device (input, processing, output and storage).
- Different types of computing devices in Coding and Robotics.
- Hardware components for input, processing and output.
- Computing devices and their purpose.
- Hardware vs Software
- Interaction between harware and software.

Networks and Communication

- Internet as example of a network.
- Information and Communication Technology (ICT) - components and real-world uses.

Figure 2-10 Digital Awareness Concepts

2.6.3.3 Digital Skills

An essential set of a range of abilities that enable individuals to effectively use digital devices, software, and platforms to perform various tasks.

Application Skills

- Use and manage applications used in Coding and Robotics (software environment).
- File and Folder management
- Input, processing and output in computing and robotics
- Use an application such as Paint to create backgrounds and sprites

Patterns and Communication

- Patterns in coding and robotics for communication, including data analysis, visualisation and conveying messages or information.

Digital Literacy

- Find, evaluate and use informaiion effectively and ethically

Figure 2-11 Digital Skills Concepts

2.7 APPROACH TO TEACHING CODING AND ROBOTICS

Coding and Robotics, as a subject, is process-driven as it focuses on Coding and Robotics processes, rather than just exit skills or products. Coding develops cognitive and critical thinking skills as it emphasises the development of knowledge, skills, strategies, and attitudes that enable learners to become more effective individuals. Coding and Robotics also supports learners to develop metacognitive skills, which include planning, developing, testing, evaluation and reflecting.

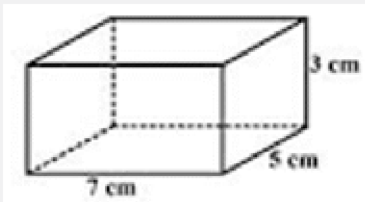
2.7.1 Problem-based Learning

Teaching and learning will follow a problem-based learning approach. Problem-based learning (PBL) is an active and learner-centred approach to learning involving several cognitive processes that aims to develop critical thinking, problem-solving, and collaboration skills. The goal of PBL is to help learners learn how to apply knowledge and skills using problems, rather than just memorising information for tests. PBL also encourages learners to ask questions and seek answers, rather than passively receiving information. It also supports the development of self-directed learning.

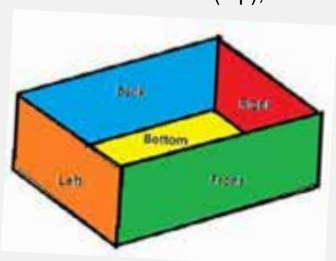
In Intermediate Phase, learners will be given small, manageable problems which they need to solve using a problem-solving process. To develop and enhance self-efficacy (the learner's belief that he/she will be able to complete the task or solve the problem), the challenge of the task or problem should match the learners' competencies.

Example of a manageable problem and algorithm development using computational thinking and the problem-solving process in Intermediate Phase:

Problem:
Calculate the surface area of a box.



Step 1: Understand the problem.
What is surface area?
Surface area is the amount of area covered by the surface of the box, i.e. all the sides (top, bottom, back, front, left and right) of the box
If one takes off the lid (top), it looks like the figure below and one can see all the sides:



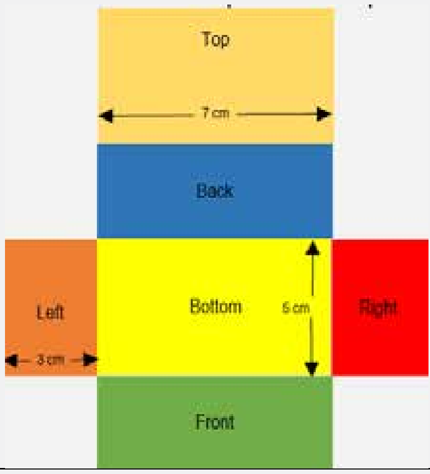
Step 2: Analyse the problem

- The box has six (6) sides.
- Each side is a rectangle.
- Calculating the area of each rectangle will provide the surface area of the box.

Step 3: Break down the problem into smaller, more manageable problems.

- **Decompose** (break down) the problem, into more familiar ones, using **abstraction.**
  - Unfold the box – this helps to break the problem into smaller parts (six (6) rectangles).
  - Each part can now be solved individually.

- Each ‘small’ solution can then be combined again to solve the ‘big’ problem.



Abstraction involves simplifying ideas by focusing on essential details. It also helps to break down a problem into manageable parts to understand the underlying structure.

Abstraction helps with **decomposition** and to simplify the problem or make it easier to understand by ignoring detail we do not need.

Step 4: Look for patterns (Pattern recognition)

The top and bottom rectangles are the same (length = 7 cm and width = 5 cm)

The left and right rectangles are the same (length = 5 cm and width = 3 cm)

The front and back rectangles are the same (length = 7 cm and width = 3 cm)

It involves finding the similarities or patterns among small, decomposed problems that can help us solve more complex problems more easily and efficiently.

Step 5: Develop a high-level solution or algorithm (abstraction)

Calculate the surface of each side (rectangle):

Step 1: Calculate area of top.

Step 2: Calculate are of front side.

Step 3: Calculate area of left side

We can now break the problem into ***three*** rather easy **sub-problems** or **main ideas (decomposition,** using **abstraction)**.

Abstraction also helps us to realise that we can ignore 3 of the rectangles / only focusing on 3 rectangles – one of each size to solve the problem.

Step 6: Detailed Algorithm (Decomposition)

1. Step 1: Calculate the area of the top.
   1.1. Multiply the area of the top by 2 (to add the area of the bottom side/rectangle).
2. Step 2: Calculate the area of the front side.
   2.1. Multiply the area of the front side by 2 (to add the area of the back rectangle).
3. Step 3: Calculate the area of the left side.
   3.1. Multiply the area of the left side by 2 (to add the area of the right rectangle).
4. Step 4: Add the areas of Step 1.1, 1.2 and 1.4 to get the surface area of the box.

Each step in the **high-level algorithm** was broken down into more specific, detailed steps, giving more detailed instructions.

Step 7: Test and Debug the Algorithm.

- Was the surface area calculated correctly?
- If the answer is yes, the problem is solved else you need to identify the error and fix the algorithm (debug).

As a next step, one can write code that will accept (as input) the measurements of the box and calculate the surface area (once variables are done)

Computational thinking is also about drawing on previous experience with tasks or problems to complete similar tasks or solve similar problems, for example, the following can flow from the problem discussed above:

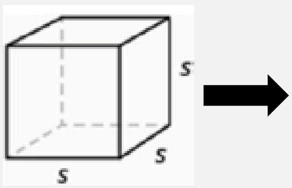
Extention

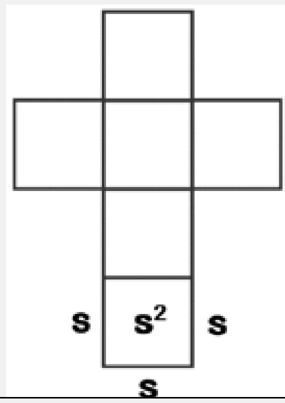
Learners can use their knowledge and experience form unfolding a box and calculating the surface area of the box to do the same for a cube, realisinng that, to calculate the surface area of a cube can be done in the same way:

Algorithm

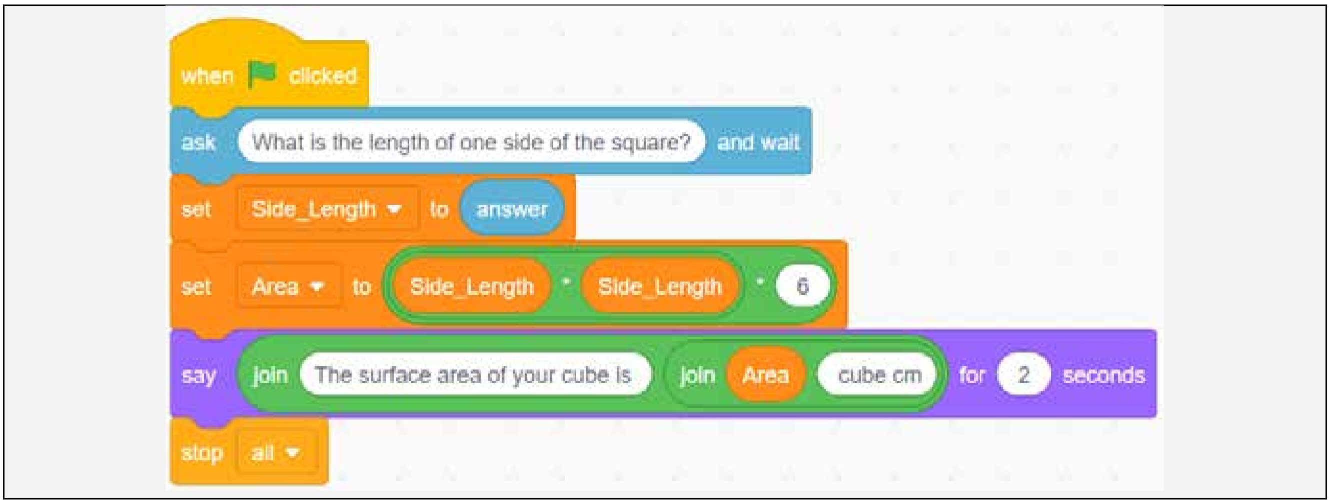
Step 1: Calculate the area of one square

Step 2: Multiply the answer (area) by 6.





Write code for the extention (towards end of Grade 6)



Generally, problem-based learning

- enables learners to develop problem solving strategies as well as subject knowledge and skills.
- enables learners to be more engaged in learning.
- stimulates critical thinking.
- promotes self-directed learning as learners generate problem-solving strategies.
- promotes metacognition as learners compare and reflect on solutions.
- assesses learning in ways which demonstrate understanding and competency.

See **Section 4.2** for problem-based learning assessment guidance.

PBL could incorporate strategies such as cooperative learning where learners work in small groups to solve a coding or robotics problem or use pair programming where learners work in pairs to solve a coding or robotics problem.

2.7.2 Cooperative Learning

Cooperative learning is an active teaching-learning strategy where learners work in small groups, they help each other learn, and in doing so, increase their joy and skills in the learning process.

Learning activities and roles are structured and overseen by the teacher, and each member of the group oversees the academic performance of the others. To successfully implement cooperative learning, leading authors in the field (David Johnson and Roger Johnson) emphasise the intentional stimulation of five basic elements (Johnson & Johnson, 2021:55-56) namely:

- **Positive interdependence**: Learners should feel like they are linked in such a way that one cannot succeed unless all in the group succeeds. Teachers should thus find ways of stimulating positive interdependence in their group activities – one possibility is giving learners different roles to fulfil; hence the group cannot move forward unless all roles are successfully fulfilled.
- **Individual accountability**: Learners should know that all will be assessed individually as well. "*The purpose of cooperative learning groups is to make each member a stronger individual in his or her right*". One way of stimulating individual accountability is by giving learners individual marks for how well they contributed to the group activity – this assessment can occur either via teacher assessment or peer assessment – by doing this, everyone will know that they cannot get a freeride during the group activity as their inputs are also individually assessed.
- **Promotive interaction**: Learners' successes are increased due to the sharing of resources, support provided, and praise and encouragement given by their group members. Teachers thus need to stimulate promotive interaction, which can be done by giving different resources to different learners. Giving learners different roles also stimulate promotive interaction.

- **Social skills**: Stimulating social skills becomes an intentional endeavour of the teacher. Teachers could provide learners with resources on how to effectively form part of a team, how to communicate well and how to resolve conflict, should it arise.
- **Group processing**: Group processing forms part of reflection during and after the group activity. Teachers can stimulate group processing by giving learners a reflection sheet or by asking them open-ended questions to stimulate reflective conversations. Questions such as: "What worked well during your group activity"? or "*Describe the best experiences and worst experiences of the group activity*".

Cooperative learning can improve the learner's performance and teaches the value of teamwork, cooperation, communication, self-denial, and initiative taking.

2.7.2.1 Implementing cooperative learning in Intermediate Phase Coding and Robotics

Example of cooperative learning activity for Intermediate phase learners on the topic of robotics (see Grade 4 (C.3)): *Execute a simple set of commands in relation to R.6, physically, on paper or with an educational tool.*

Task: Determine where a robot (simulated by one of the learners) will end after executing a set of instructions, including at most nine steps, provided in an algorithm.

Divide the class into groups of four. Two learners could take on the roles of **instructor** and **interpreter** respectively and the other two learners the roles of **robot** and **debugger**.

- **Instructor:** Reads out the steps from the algorithm
- **Interpreter**: Puts steps from algorithm into "layman's English" /Explain steps in plain English
- **Robot**: Executes the steps from the interpreter
- **Debugger**: Evaluates the movement of the robot to determine whether it executed steps correctly.

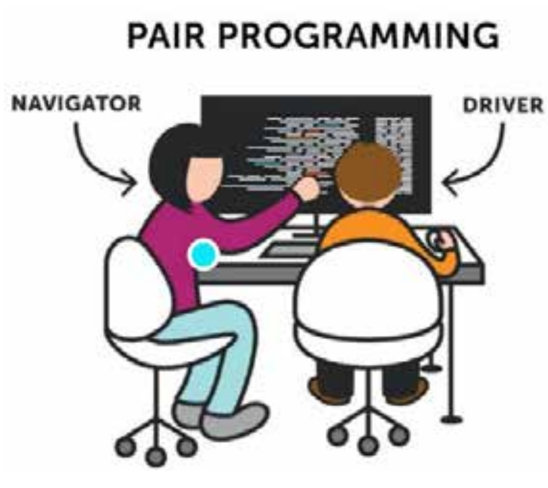
Tools that can be used to develop the algorithm: pen-and-paper, coding cards (e.g. Tanks) for algorithm and interpretation, then, code algorithm using blocks from block-based coding platform, implement, test and debug in block-based coding environment.

Refer to Annexure B for cooperative learning assessment guidance.

Pair programming could also be used as a cooperative teaching and learning strategy to solve programming problems.

2.7.3 Pair Programming

Pair programming is a pedagogical approach that involves two learners working together on one computer or one piece of paper to complete a shared goal/task. It emanates from the programming industry yet has proven to be successful even at school level. One of the learners fulfils the role of the "*driver*" while the other learner fulfils the role of the "*navigator*".



The driver is the learner who may use the computer and handles the keyboard, or draws on the paper and handles the pencil, whereas the navigator is the learner who utilises the resources, and reviews the driver's work throughout, providing feedback and suggestions to the driver, pointing out errors and asking questions of the teacher. Pair programming is a collaborative effort that involves a lot of communication, discussion, and problem-solving.

Although pair programming can be implemented as a collaborative “unstructured” pair activity, it is best to stimulate the five basic elements of cooperative learning as described above, when implementing pair programming in the classroom.

It also appears particularly promising in situations where there are not enough computing devices for learners to work individually, as well as for increasing learning and engagement with technology by learners with limited device experience. It is also suggested that learners show higher confidence when programming in pairs. It allows learners to share knowledge and learn from each other, thereby improving the quality of the learning engagement.

2.7.3.1 Implementing pair programming in Intermediate Phase Coding and Robotics

Example of pair programming activity for Intermediate phase on the topic of Coding (C.1 and C.2):

Apply computational thinking skills to develop a set of logical instructions to solve a problem.

Learners are divided into pairs. Learners should draw a square using a set of instructions. One learner fulfils the role of “driver” and the other “navigator”.

- **Driver** – The learner acting as the driver will be the one completing the steps in a block-based program and/or unplugged on a piece of paper.
- **Navigator** – The learner acting as the navigator may consult the textbook and/or other resources. The learner may also ask the teacher for help.

Learners need to find a way to draw a rectangle using a set of instructions. This implies having the drawing tool “turn” several degrees and moving forward several pixels/steps. Learners should be able to first work this out by “directing” each other and then put these instructions over into algorithm.

Note:
The teacher may swop the learners’ roles as the activity progresses to ensure that both learners have a chance to fulfil each role. You may also ask any one of the learners to present their work to the class. This ensures that both learners feel a need to engage and gives more learners an opportunity to practice communication skills.

2.7.4 Deliberate Practise

A subject such as Coding and Robotics not only requires thinking skills, but also requires focused teaching and ample practise. This practise should, however, be purposeful, well thought through with gradual increase in complexity.

The curriculum is designed to encourage deliberate practise, as competencies are repeated within and across grades. The concept of deliberate practise is particularly focused on skill acquisition and development and is key in the development of competency and expertise in subjects such as coding.

Deliberate practise is a specific type of practise that involves setting specific goals, receiving feedback, and making focused efforts to acquire and improve skills and performance. It is not simply repeating skills over-and-over again, but rather adjusting to improve competencies as well as gradually adding additional competencies that lead to mastery. It therefore involves purposeful repetition, feedback-driven metacognition, and extension to improve performance (Ericsson, 2008; Deans for Impact, 2016; Ericsson *et. al*., 2018).

In terms of extension, deliberate practice involves extending the amount of time spent practising, adding new features, and increasing the complexity of tasks. The goal is to push beyond one's comfort zone to achieve growth and improvement.

2.7.5 Science of Learning

Science of Learning, a multidisciplinary field combines research from cognitive psychology, neuroscience, educational psychology, and other related disciplines to understand how people learn. It also aims to identify the most effective teaching and learning strategies based on empirical evidence that has been shown to improve long-term retention of information and enhance learning outcomes.

Learning is an iterative process that requires that one continually revisits what one has learned earlier, update it, and connect it with new knowledge. Learning always builds on a store of prior knowledge and is the residue of thought. New learning requires a considerable amount of practise and meaningful connections to existing knowledge. Learning, therefore, requires learners thinking (Brown *et al*., 2014; Dereck Bok Center, Harvard University, 2023).

Science of learning includes the following learning strategies (*Weinstein et* al., 2018):

- **Retrieval practice**: Bringing learned information to mind from long-term memory.
- **Spaced practice**: Spreading learning activities out over time/reviewing previously learned information at gradually increasing intervals.
- **Interleaving**: Switching between topics while learning.
- **Examples**: When learning abstract concepts, illustrating them with various examples or experiences.
- **Dual coding**: For example, combining visuals with text.
- **Elaboration**: Classroom discussions that require learners to relate new material to what they already know and to recall previously learned information, including asking *why* and *how* questions with learners explaining in their own words.
- **Interactive activities**: Engage actively with learning material using activities that require one to retrieve (recall) previously learned information.

2.8 LINKING CODING AND ROBOTICS WITH OTHER SUBJECTS

Coding and Robotics concepts can be linked to Language, Mathematics, Natural Science and Technology and Life Skills in the Intermediate Phase. These cross-cutting concepts should therefore be integrated into Coding and Robotics to enhance the learning experience.

For example, coding often involves mathematical concepts such as logic, arithmetic, and geometry whilst Robotics combines coding with principles of physics, engineering, and materials science, highlighting the interdisciplinary nature of digital concepts and skills. Other examples:

Algorithms involve sequencing and summarising in literacy and breaking down complex problems into simpler steps in mathematics.

Modularity: Involves breaking down tasks into manageable units in computer science, while in mathematics, it involves breaking down a complex problem into smaller, manageable parts.

Control structures: Determine how a set of instructions are executed within a program, while heuristic thinking in mathematics involves using logical thinking and trial and error to solve problems.

Coding and natural language: The process of learning to code is also often likened to language acquisition, as learners progress through six distinct stages of understanding. These stages bear close resemblance to the stages of literacy development.

Design: Designing robotics artefacts links to aspects of Creative Arts.

Digital concepts: Aspects such as the impact of technology and being a digital citizen, links to Life Skills.

By developing these skills in Coding and Robotics, learners can develop habits of mind and analytical thinking that will be valuable in all other subjects.

2.9 Time Allocation

In Intermediate Phase, 2 hours per week (20 hours per term) is allocated for Coding and Robotics.

The following table provides the time allocation as a percentage of the total available time per term:

Table 2-3 Time allocation for Intermediate Phase Coding and Robotics

	Grade 4	Grade 5	Grade 6	Grade 7	Grade 8	Grade 9
Coding	50	50	50	50	45	45
Robotics	25	25	30	30	35	35
Digital Concepts	25	25	20	20	20	20

Available time should be allocated as indicated by the percentages in the table above.

Note:

Sections 2.12.1 (coding content) 2.12.2 (robotics content) and 2.12.3 (digital concepts content) are linked and support each other. Various competencies across the three strands can therefore be linked and dealt with in an integrated fashion. Section 3 (unpacking of the content) provides examples and notes and suggests pedagogical approaches.

2.10 Resources required to offer Coding and Robotics in Intermediate Phase

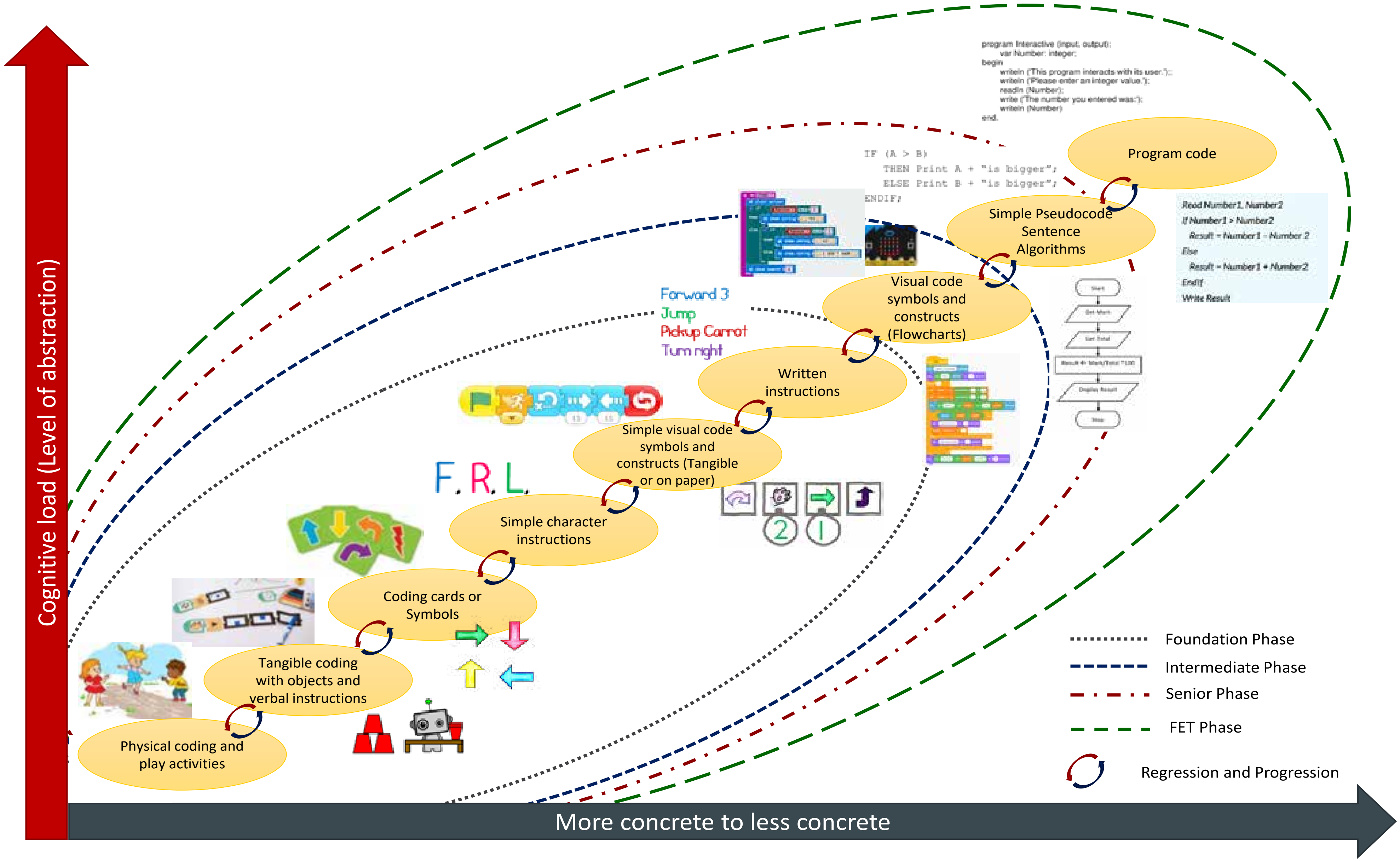


Figure 2.12: Programming resources for Coding and Robotics

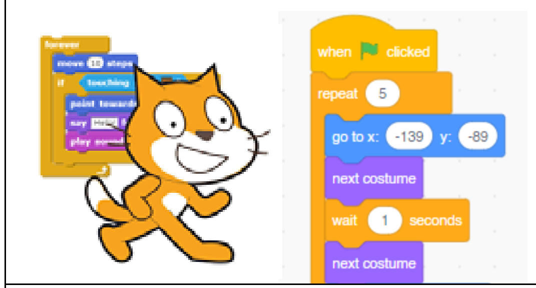
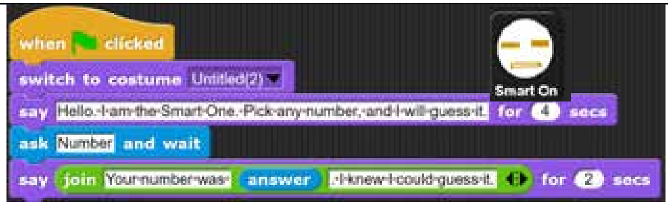
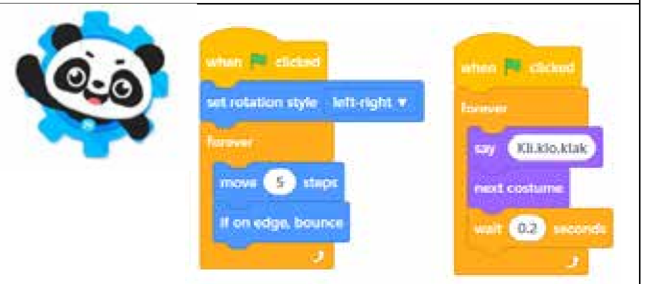
2.10.1 Coding Resources

Refer to Figure 2.12:

In intermediate phase, learners will follow a block-based coding approach. It allows users, especially novices and kids, to create programs using visual elements like blocks and symbols. In this type of environment, users can drag and drop various blocks representing commands and snap them together like puzzle pieces to create their programs. It helps learners to the learn the foundational concepts and principles of coding without getting overwhelmed by the intricacies of text-based syntax and it minimises errors (such a formatting, punctuation, or spelling mistakes, semantics that could discourage learners) associated with the complex syntax of text-based environments. By abstracting away the textual complexities, it therefore reduces cognitive load and allows learners to focus on the problem and the foundational coding concepts as well as the underlying logic of their programs, rather than too much mental effort on the code syntax. It therefore serves as an effective steppingstone for beginners to develop their problem-solving and programming skills before transitioning to more advanced coding environments. Research also suggests that teaching computational thinking with block-based coding, learners (1) achieved substantial learning gains in algorithmic thinking skills, (2) were able to transfer their learning from block-based to a text-based programming context, and (3) achieved significant growth toward a more mature understanding of computing as a discipline (Grover, Pea & Cooper, 2015).

> "Computer programming is a highly cognitive skill, which requires mastery of multiple domains, and is acknowledged as being difficult to learn, making it essential to take into account the cognitive loads (CLs) imposed on learners, as well as their abilities to absorb this knowledge during the teaching and learning process". *Berssanette & de Francisco (2022).*

Examples include:

Scratch	Snap	mBlock
		

Note

A program is a sequence of symbols that specifies a computation.

A programming language is a set of rules that specify which sequences of symbols constitute a program, and what computation the program describes.

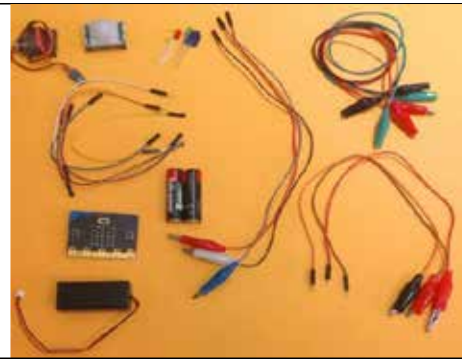
A programming language is an abstraction mechanism. It enables a programmer to specify a computation abstractly, and to let a program (usually called an assembler, compiler or interpreter) implement the specification in the detailed form needed for execution on a computer (Ben-Ari, 2006)

Where learners struggle, physical coding or coding cards could be used as support and remediation.

2.10.2 Robotics resources

The tables below list resources used in some example projects throughout this document. The list is not exhaustive.

Refer to Section 2.15.3:

		
Plastic gears, fans, base plates (Lego plates) Rubber bands Fans Pully's Connectors Pipe cleaners Googley eyes Plastic wheels Screws and nuts Pipe cleaners	Simple switches Wire Straws String or Cable ties Double sided tape Small pieces of wooden blocks (available from arts and craft shops) Plastic bottle caps (for wheels) Servo Motor (E.g., SG90)	PIR Motion Sensor (E.g., HC-SR501) Crocodile to Male pin connectors Crocodile to Female pin connectors Crocodile clips (Clips at both ends) Batteries LEDs Connection wire with pins Foil Microcontroller

For robotics, learners will follow a block-based coding approach. Block-based coding helps learners to the learn the basics and foundational concepts of coding in a visual, syntax-free environment. Visualised coding minimises errors associated with the complex syntax of text-based environments. It reduces cognitive load and allows learners to focus on the coding problem and the foundational coding concepts.

2.10.3 Digital Concepts Resources

- Sample technologies and components (e.g., mobile phone, tablet, laptop (with input and output devices, etc.)
- Pictures of computing devices, input devices, output devices
- Simple diagrams of networks, etc.

2.11 Overview of Intermediate Phase Coding and Robotics

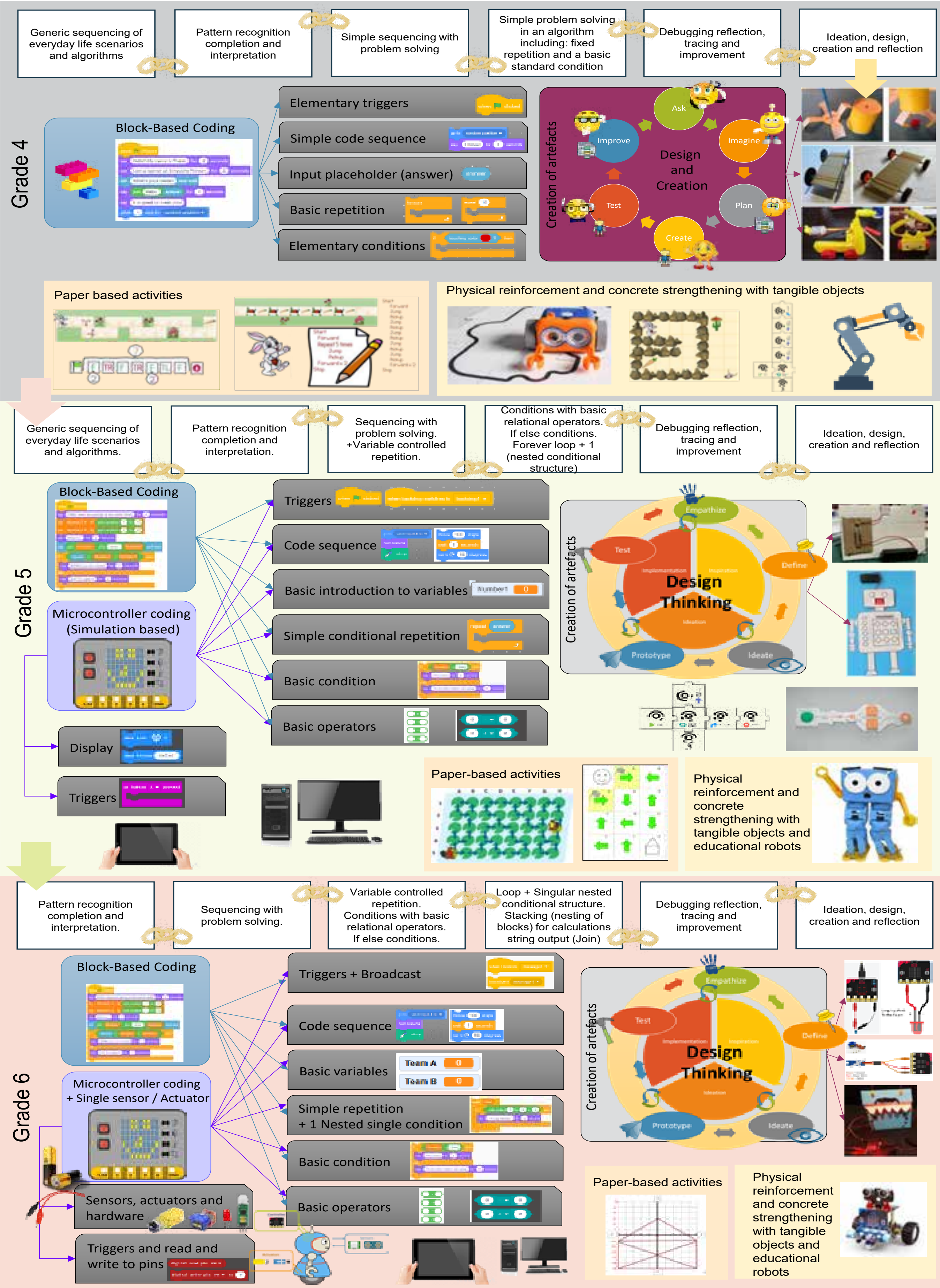


Figure 2.10 Coding & Robotics Overview Grade 4 - 6

2.12 Focus of Content Areas

2.12.1 Coding

Table 2-4: Coding content focus and progression

<table>
<tr><th>Competency</th><th>Grade 4
(beginner level)</th><th>Grade 5
(advanced beginner level)</th><th>Grade 6
(moderate level)</th></tr>
<tr><td rowspan="3">C.1
Apply computational thinking skills to develop a set of logical instructions to solve a problem.</td><td colspan="3">Progression in problem-solving mostly lies in gradually increasing the scope and complexity of problems.</td></tr>
<tr><td>Using foundational problems:
• Develop a set of logical instructions to solve a foundational problem that includes (where appropriate):
- sequences of commands
- single repetition
- simple conditional constructs
Refer to Table 2-7
• Trace, evaluate, correct or complete a set of logical instructions (algorithm)
(link to C.2, C.3, C.4, C.5, C.6 and C.7)</td><td>Using basic problems:
• Develop a set of logical instructions to solve a basic problem that includes (where appropriate):
- sequences of commands
- single repetition
- simple conditional constructs
Refer to Table 2-7
• Trace, evaluate, correct or complete a set of logical instructions (algorithm)
(link to C.2, C.3, C.4, C.5, C.6 and C.7)</td><td>Using simple problems:
• Develop a set of logical instructions to solve a simple problem that includes (where appropriate):
- sequences of commands
- single repetition
- simple conditional constructs
Refer to Table 2-7
• Trace, evaluate, correct or complete a set of logical instructions (algorithm)
(link to C.2, C.3, C.4, C.5, C.6 and C.7))</td></tr>
<tr><td colspan="3">Computational thinking is infused and used in all aspects of coding/problem solving activities as follows:
Use abstraction to simplify complex problems by reducing it to its most essential components. It helps to gain a deeper understanding of their underlying structure and develop more effective solutions. It allows you to focus on the essential aspects while minimizing distractions from irrelevant details. Abstraction is also used to make a visual representation of solution.
Use decomposition to help to simplify complex problems by breaking them down into manageable parts, that allows one to address each component individually. It enables a systematic approach to problem-solving and enhances ones understanding of the problem's structure and relationships.
Use pattern recognition to leverage existing knowledge and experiences to identify meaningful regularities in data or situations. Recognising patterns, helps to gain insights, make predictions, apply them to new problems, and develop more effective problem-solving strategies.
Use algorithmic thinking to develop a series of precise, logical steps or instructions (algorithm) to accomplish a task or solve a problem in an organised and methodical manner.</td></tr>
<tr><td>C.2
Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.</td><td>Working with a coding solution (set of logical instructions) for foundational coding problems:
• Translate an elementary coding solution (set of instructions/algorithm) into programming code (e.g. block-based coding instructions / coding cards, etc.).
• Use code (symbols/ blocks/written statements to represent actions and operations to accomplish a particular task/solve a programming problem.
• Group instructions/code blocks to represent repetition (or a statement indicating repetition)
(Done in relation to C.1)</td><td>Working with a coding solution (set of logical instructions) for basic coding problems:
• Translate a basic coding solution (set of instructions/algorithm) into programming code (e.g. block-based coding instructions / Coding cards, etc.).
• Use code (symbols/ blocks/written statements to represent actions and operations to accomplish a particular task/solve a programming problem.
• Group instructions/code blocks to represent repetition (or a statement indicating repetition)
(Done in relation to C.1)</td><td>Working with a coding solution (set of logical instructions) for simple coding problems:
(Translate a simple coding solution (set of instructions/algorithm) into programming code (e.g. block-based coding instructions/Coding cards, etc.).
• Use code (symbols/ blocks/written statements to represent actions and operations to accomplish a particular task/solve a programming problem.
• Group instructions/code blocks to represent repetition (or a statement indicating repetition)
(Done in relation to C.1)</td></tr>
<tr><td>C.3
Interpret and execute a given symbolic or written set of commands</td><td>Using foundational coding problems:
• Execute a set of commands using unplugged activities (physically, on paper, coding cards) or an educational tool (e.g. block-based coding software).
Done in relation to C.1 and C.2
• Determine the output of a given algorithm (set of commands) or of given program code (e.g. trace block-based commands) to determine the output or explain what the code/program does).</td><td>Using basic coding problems:
• Execute a set of commands using unplugged activities (physically, on paper, coding cards) or an educational tool (e.g. block-based coding software).
Done in relation to C.1 and C.2
• Determine the output of a given algorithm (set of commands) or of given program code (e.g. trace block-based commands) to determine the output or explain what the code/program does).</td><td>Using simple coding problems:
• Execute a set of commands using unplugged activities (physically, on paper, coding cards) or an educational tool (e.g. block-based coding software).
Done in relation to C.1 and C.2
• Determine the output of a given algorithm (set of commands) or of given program code (e.g. trace block-based commands) to determine the output or explain what the code/program does).</td></tr>
</table>

C.4 **Debug a given symbolic or written set of instructions.**	• Reinforce reading and understanding a foundational problem. • Interpret/execute/trace a given set of commands to determine correctness of the solution/if the correct output is achieved. • Complete an incomplete a set of commands provided to solve a given foundational problem. • Inspect/trace a foundational coding solution (set of commands) for an error or errors and correct if necessary. (Unplugged and Plugged) Done in relation to C.1, C.2, C.3 and C.4	• Reinforce reading and understanding a basic problem. • Interpret/execute/trace a given set of commands to determine correctness of the solution/if the correct output is achieved. • Complete an incomplete a set of commands provided to solve a given basic problem. • Inspect/trace a basic coding solution (set of commands) for an error or errors and correct if necessary. (Unplugged and Plugged) (Done in relation to C.1, C.2, C.3 and C.4	• Reinforce reading and understanding a simple problem. • Interpret/execute/trace a given set of commands to determine correctness of the solution/if the correct output is achieved. • Complete an incomplete a set of commands provided to solve a given simple problem. • Inspect/trace a simple coding solution (set of commands) for an error or errors and correct if necessary. (Unplugged and Plugged) Done in relation to C.1, C.2, C.3 and C.4
C.5 **Evaluate a given solution towards potential improvement.**	Using foundational coding problems: • Reflect and report on a given solution by asking the following questions (critical thinking): - What happened? - Why has it happened? - What can be learnt? - How can the solution be improved? • Inspect a set of commands (algorithm/program) and reflect to improve it or provide a better alternative (e.g. reducing the number of steps/instructions using a loop for repetitive steps/patterns Link to C.6 and C.7.	Using basic coding problems: • Reflect and report on a given solution by asking the following questions (critical thinking): - What happened? - Why has it happened? - What can be learnt? - How can the solution be improved? • Inspect a set of commands (algorithm/program) and reflect to improve it or provide a better alternative (e.g. reducing the number of steps/instructions using a loop for repetitive steps/patterns Link to C.6 and C.7	Using simple coding problems: • Reflect and report on a given solution by asking the following questions (critical thinking): - What happened? - Why has it happened? - What can be learnt? - How can the solution be improved? • Inspect a set of commands (algorithm/program) and reflect to improve it or provide a better alternative (e.g. reducing the number of steps/instructions using a loop for repetitive steps/patterns Link to C.6 and C.7
C.6 **Recognise and interpret patterns in symbolic sets of data or visualisations.**	• Identify a foundational pattern (e.g. a pattern in coding instructions, numbers, symbols, blocks, characters, sequences, etc) • Interpret, explain and complete/ extend a foundational pattern (describe the pattern rule, use the pattern rule to complete/extend the pattern or make predictions) Link to C.1 and C.2	• Identify a basic pattern (e.g. a pattern in coding instructions, numbers, symbols, blocks, characters, sequences, etc) • Interpret, explain and complete/ extend a basic pattern (describe the pattern rule, use the pattern rule to complete/extend the pattern or make predictions) Link to C.1 and C.2	• Identify a simple pattern (e.g. a pattern in coding instructions, numbers, symbols, blocks, characters, sequences, etc) • Interpret, explain and complete/ extend a simple pattern (describe the pattern rule, use the pattern rule to complete/extend the pattern or make predictions) Link to C.1 and C.2
C.7 **Create or complete a pattern to represent a data set.**	• Complete a foundational pattern that is part of a data set or programming solution. • Create a foundational pattern to form part of a data set or a programming solution. Done in relation to C.6 Link to C.1 and C.2	• Complete a basic pattern that is part of a data set or programming solution. • Create a basic pattern to form part of a data set or a programming solution. • Generalise a basic pattern based on the pattern rule. Done in relation to C.6 Link to C.1 and C.2	• Complete a simple pattern that is part of a data set or programming solution. • Create a simple pattern to form part of a data set or a programming solution. • Generalise a simple pattern based on the pattern rule. • Incorporate the generalised pattern as part of a programming solution. Done in relation to C.6 Link to C.1 and C.2

Note

Linked competencies can be grouped/done together within one lesson/activity where appropriate.

2.12.2 Robotics

Table 2-5: Robotics content focus and progression

Outcome	Grade 4 (beginner level)	Grade 5 (advanced beginner level)	Grade 6 (moderate level)
R.1 **Explain what a robot is in simple terms.**	Provide a foundational definition of a robot that includes elementary features and purpose. Link to R.2 and R.3	Provide a basic description of a robot in terms of (extend from Grade 4): • attributes • purpose • the origin of the term • concept of a controller (extend from Grade 4) • contexts in which they operate. • evolution of robots (automation – mechanical) Link to R.2 and R.3	Provide a simple description of a robot in terms of (extend from Grade 5): • definition (what it is) • purpose • the contexts that they operate in. • concepts regarding the relationship between the composition of a robot - basic parts (sensors, controllers, actuators, power source) • evolution of robots / advancements of robots (also link to the concept of AI – elementary reference to automatic decisions) Link to R.2 and R.3
R.2 **Identify different types of robots.**	Provide a foundational overview of different types of robots and their uses (virtual vs physical robots) Provide a foundational description to distinguish between a virtual and physical robot. Link to R.1 and R.3	Provide a basic overview of different types of robots and their uses (range: industrial, service, educational, medical, exploration) Give a basic description of different types including a basic description of their composition and purpose (range: mobile, Industrial, medical, education and service) Link to R.1 and R.3	Provide a simple overview of different types of robots and their uses (range: industrial, service, educational, medical, exploration) Classify robots in terms of their description, attributes and uses (range: industrial, service, educational, medical, mobile, exploration, autonomous and remote controlled) Link to R.1 and R.3
R.3 **Outline the different components of a robot**	Provide a foundational reference to the basic components of a robot and their purpose. (range: motors and mechanics for movement, sensors for observation, and actuators to respond, processor, and power source) Link to R.1, R.2 and R.4 – R.7	• Outline the basic components of a robot, with a basic explanation of the purpose of each (range: sensors, communication, grippers and attachments, actuators, controllers, power sources, structural components) • Present a basic diagrammatical outline of a robot (showing the various components) • Present a basic outline of an educational controller and its parts (e.g. buttons, sensors, LEDs, sound) • Present a basic understanding that robots are controlled by controllers, and act based on sensory or triggered input. • Present a basic outline of how a robot is coded to perform tasks. Link to R.1, R.2 and R.4 – R.7	• Outline the basic components of a robot, with a simple explanation of the purpose of each (range: sensors, communication, grippers and attachments, actuators, controllers, power sources, structural components) • Outline, at a simple level, how sensors are used in basic robots (different sensors and their purpose) (Range: Ultrasonic, microphone, Motion sensor). • Present a simple outline of an educational controller and its parts (e.g. buttons, sensors, LEDs, sound, etc) (Extend on sensors e.g. sound sensors) • Present a simple understanding that robots are controlled by controllers, and act based on sensory or triggered input. • Provide a simple outline of the process of sensing, perception, cognition, acting (in terms of how a robot interacts with the real world) • Provide a simple` outline of how a robot is controlled. Link to R.1, R.2 and R.4 – R.7
R.4 **Present an understanding of**	At a foundational level, compare the role of robots and people in the real-world doing the same task. (range: time	• At a basic level, compare the role of robots and people in the real-world doing the same task expanded with a	• At a simple level, compare the role of robots and people in the real-world doing the same task.

how robots affect the world.	saving, assistance, education, entertainment) Link to R.1 – R3 and R.5 - R.7	short description. • Provide a basic outline of the benefits and risks associated with the use of robots. • Basic explanation of what software for robots / embedded system are used for, is given that reference for specific concepts that robots can be programmed to react to their environment. Incorporates elements of R.1.5 and R.2.5 • Present a basic definition of AI and its relationship to the field of robotics Link to R.1 – R3 and R.5 - R.7	• Provide a simple outline the benefits and risks associated with the use of robots. • Present a simple outline the ethical considerations related to the use of robots. • Simple explanation of what software for robots / embedded system are used for, is given that reference for specific concepts that robots can be programmed to react to their environment. Incorporates elements of R.1.5 and R.2.5 • Simple outline of how AI is applied in the field of robotics Link to R.1 – R3 and R.5 - R.7
R.5 **Design a simple artefact based on a set of design specifications.**	• Provide a foundational outline of design thinking (inspire, ideate (imagine), implement). • Design foundational robot artefacts using the design thinking process. Link to R.1 – R.4 and R.6, R.7	• Provide a basic definition of design thinking and the design thinking process. • Provide a basic outline of the design thinking process (steps). • Design basic robot artefacts using the design thinking process/ Link to R.1 – R.4 and R.6, R.7	• Provide a simple definition of design thinking and of the design thinking process. • Provide a simple outline of the design thinking process (steps) • Design simple robot artefacts using the design thinking process. • Provide a simple outline of the relationship between the concept of hydraulics and robots. • Provide a simple introduction to an open and closed circuit. • Provide a simple introduction to a basic, single pin (Read) • Use On pin - Pressed • Basic introduction to: - an LED - a PiR sensor (Sensor as input trigger) - a Servo motor (Servo motor as an actuator) Link to R.1 – R.4 and R.6, R.7
	Design thinking is infused and used when creating robotic artifacts as follows: **Empathise**: Ask questions to find out what the problem is and to identify challenges related to the problem as well as to identify ways to solve the challenges **Define**: Specify the detail of the problem **Ideate**: Imagine and brainstorm different ideas for solving the problem and choose the best idea **Prototype (Plan and design)**: Draw a simple picture (abstraction) and write down the material you will need. Then write down step-by-step instructions (algorithm) for implementing the idea. **Test (Create/implement, test, reflect and improve):** Follow the design (picture) and plan (algorithm) and build the artefact. Then test it to see if it works and correct/improve where necessary.		
	The progression mostly lies in the gradual increase in scope and complexity of artefacts.		
R.6 **Mimic the operations of a robot**	Use a ***simulated*** environment (such as Scratch or any other free educational software tool) to mimic the operations of a robot: • Align coding concepts to be used with the coding concepts covered and mastered in the coding section. • Include role play (acting out), and tangible activities. • Includes the use of appropriate paper-based exercises. Link to R.3, R.5 and R.7	Use a ***simulated*** environment (such as MakeCode (for micro: bit)) or any other free educational software tool) to mimic the operations of a robot: • Align coding concepts to be used with the coding concepts covered and mastered in the coding section. • Include role play (acting out), and tangible activities. • Include the use of appropriate paper-based exercises. Link to R.3, R.5 and R.7	Use a ***simulated*** environment (such as MakeCode (for micro: bit) or any other free educational software tool) with a physical microcontroller (board) to mimic the operations of a robot: • Align coding concepts to be used with the coding concepts covered and mastered in the coding section. • Includes role play (acting out), and tangible activities. • Includes the use of appropriate paper-based exercises.

	In addition, schools can also opt to use tangible tools and educational robots (OPTIONAL) for reinforcement.	In addition, schools can also opt to use tangible tools and educational robots (OPTIONAL) for reinforcement.	Link to R.3, R.5 and R.7 In addition, schools can also opt to use tangible tools and educational robots (OPTIONAL) for reinforcement.
	The scope and complexity are gradually increased in relation to the coding features, operations and structures (code blocks/coding constructs – coding knowledge and skills) introduced per term per year as well as in terms of the complexity of the problem. **Refer to table 2-8**		
R.7 **Create, test, and execute a set of robotic instructions.**	Using foundational problems: • Develop foundational solutions to solve a specific problem. • Translate solution instructions (algorithms) into code (using virtual robots with instructions in a tangible or non-tangible coding environment (software) or using physical educational robotic tools or both). • Implement, test, modify and/or improve foundational solutions. (Link to C.1 to C.5 as well as R.5 to R.6)	Using basic problems: • Develop basic solutions to solve a specific problem. • Translate solution instructions (algorithms) into code (using virtual robots with instructions in a tangible or non-tangible coding environment (software) or using physical educational robotic tools or both). • Implement, test, modify and/or improve basic solutions. (Link to C.1 to C.5 as well as R.5 to R.6)	Using simple problems: • Develop simple solutions to solve a specific problem. • Translate solution instructions (algorithms) into code (using virtual robots with instructions in a tangible or non-tangible coding environment (software) or using physical educational robotic tools or both). • Implement, test, modify and/or improve simple solutions. (Link to C.1 to C.5 as well as R.5 to R.6)
	The scope and complexity are gradually increased in relation to the coding features, operations and structures (code blocks/coding constructs) introduced per term per year as well as in terms of the complexity of the problem.		

Note
Linked competencies can be grouped/done together within one lesson/activity where appropriate.

Note
Learning to walk to school independently involves carefully considering each step at first. Sidewalks are identified, crosswalks are used appropriately, and perhaps even a song is sung to remember the route.
However, with repeated walks to school, the process becomes progressively easier. Landmarks like houses and shops become recognizable. Stopping and checking for cars becomes automatic, and foot placement requires less conscious thought. The experience transforms into a game-like activity.
Similarly, problem-solving skills are developed. Initial attempts may necessitate assistance. But through continued practice, proficiency increases. Patterns are identified, and plans are formulated independently, akin to navigating a puddle on the sidewalk.
Ultimately, these skills become ingrained. Their use may even go unnoticed. Just like walking to school, the process becomes second nature. Repeated practice, however, leads to continued improvement in critical thinking and problem-solving abilities, regardless of the situation encountered.

2.12.3 Digital Concepts

Table 2-6: Digital Concepts content focus and progression

Competency	Grade 4	Grade 5	Grade 6
D.1 **Outline the concept of technology and the purpose of information technology (IT).**	• Present a foundational explanation of what technology is. • Present a foundational explanation of what information technology is. • Relate the concept of technology and information technology to that of a tool. • Identify the information technology used in a specific real-world scenario (home/school environment) and explain the purpose. • Identify examples of information technology and relate their use and purpose to everyday life. Link to D.3, D.4 and D.5	• Provide a basic explanation of what a computer in the context of information technology is. • Relate the concept of computers to that of an IT tool. • List examples of computers and relate their use and purpose to everyday life. • Understand the purpose of information technology and its role in general. • Identify the information technology used in a specific real-world scenario (e.g. entertainment, shopping) and explain the purpose. Link to D.3, D.4 and D.5	• Provide a simple explanation of what a computer in the context of information technology is. • Relate the concept of computers to that of an ICT tool. • Describe examples of computers and relate their use and purpose to everyday life. • Compare and evaluate the role of information technology in two different contexts (e.g. education, shopping, and entertainment) and discuss advantages and disadvantages. Link to D.3, D.4 and D.5
D.2 **Recognise that he or she is living as citizens in a digital world.**	• Understand what the digital world is. • Provide a foundational understanding of a digital world and a digital citizen. • Understand how to use technology and computers in the classroom responsibly. • Recognise the dangers of the online environment (online predators, addiction, and distraction). • Provide a foundational understanding of - Cyberbullying and how to deal with it. - Reason for using passwords/ pins (security). - The concept and dangers/risks of sharing information like personal information, usernames, and passwords. - A digital footprint Link to D.6	• Give a basic explanation of the digital world all around us. • Provide a basic description of a digital world and digital citizenship. • Understand how to use technology and computers in the classroom responsibly and when to report unsuitable use, unauthorised access of content and/or contact. • Understand the dangers of the online environment (online predators, addiction, false information) • Provide a simple understanding of - Cyberbullying and how to deal with it. - Reason for using passwords/ pins (security). - The concept and dangers of sharing information like personal information, usernames, and passwords/pins. - A digital footprint Link to D.6	• Provide a simple explanation of the digital world all around us. • Explain digital citizenship. • Explain how to use technology and computers in the classroom responsibly and when to report unsuitable use, unauthorised access of content and/or contact. • Understand ethical issues and dangers associated with the use of information technology, including privacy, security, copyright, false information and inappropriate content. • Provide guidelines on how to manage: - Cyberbullying - Passwords/pins (security). - Sharing of personal information. - Digital footprints Link to D.6
D.3 **Demonstrate an understanding of the concept of a computing device.**	• Provide a foundational definition of a computing device, including concepts of input, processing, output, and storage. • Identify common computing devices, e.g., tablet, PC and what they are used for. (Link to D.1 and D.2) • Understand the concepts of hardware and software (“apps”). (Link to C.2)	• Provide a basic description of a computing device, including the concepts of input, processing, output, and storage. • Distinguish between the concepts of hardware and software. • Provide a list of common computing devices and describe what they are used for. • Provide a list of common apps found on devices (e.g., WhatsApp) (Link to D.1 and D.2) • Describe and demonstrate the concept of working in and navigating an application (app) (Link to C.2) • Identify the software (‘apps’) one can use on the devices identified	• Explain what a computing device, is in terms of input, processing, output, and storage. • List common input, output, and storage devices. • Explain the purpose and role of hardware (as input, processing, storage, and output devices) and software as a list of instructions (apps) that the computer can follow. • Describe the common computing devices and describe their input, output, and storage devices. (Link to D.1 and D.2) • Identify the software (‘apps’) one can use on the devices and the basic function/purpose of those

			(e.g., block-based coding app to write computer programs). • Explain and demonstrate the concept of working in and navigating an application (app) (Link to C.2)
D.4 **Identify the common uses of ICT in the real world.**	• Provide a foundational definition of what ICT is (inclusion of the concept of 'communication' in 'IT' that allows people to interact in the digital world). • Identify everyday uses of ICTs, e.g., mobile phones (communication) (Link to D.1, D.2, D.3 and D.4)	• Provide a basic definition of what ICT is (inclusion of 'communication' in 'IT' that allows people to interact in the digital world). • Provide a basic understanding of everyday uses of ICTs, e.g., computers connected using a network. • Basic understanding of a network (e.g. school network / entertainment / shopping) (Link to D.1, D.2, D.3 and D.4)	• Provide a simple definition of what ICT is (ICT is an umbrella term that includes any communication devices and systems). • Provide a simple explanation of everyday uses of ICTs, e.g., smart TV (entertainment), point-of-sales (business). • Simple understanding of a network; (devices connected in e.g. shopping, cellular, education). (Link to D.1, D.2, D.3 and D.4)
D.5 **Differentiate between the components of an ICT system.**	• Provide a foundational understanding of an ICT system (includes hardware, software (computing devices and communication) – a foundational understanding that additional hardware/technology is required to enable communication ('form networks'). (Link to D.1, D.2, D.3 and D.4)	• Provide a basic understanding of an ICT system (includes hardware, software (computing devices) and communication (concept of network)) – a basic understanding that additional hardware/technology is required to enable communication ('form networks')). (Link to D.1, D.2, D.3 and D.4)	• Provide a simple understanding of an ICT system (includes hardware, software (computing devices) and communication (concept of network) and people) – a simple understanding that additional hardware/technology is required to enable communication ('form networks')) (Link to D.1, D.2, D.3 and D.4)
D.6 **Explain how the adaptation of technology impacted the world we work and live in.**	• Provide a foundational understanding of how technology impact how we interact with others. (Link to D.2, D3, D.4 and D.5).	• Provide a basic understanding of how technology impacts the following: - Interaction with others - Communication - False information/Fake news (Link to D.2, D3, D.4 and D.5).	• Provide a simple understanding of how technology impacts the following: - Interaction with others - Access to information - Entertainment (movie/audio streams, music instruments, games) - False information/Fake news (including fact checking) (Link to D.2, D3, D.4 and D.5).
D.7 **Present a basic understanding of the concept of input processing and output.**	• Present a foundational understanding that input results in some form of output. • Illustrate through a foundational activity how input results in some form of output (e.g. open & close programs). • Present a foundational understanding of the concept that processing takes place between input and output. • Provide a foundational understanding that different forms of input result in different actions/outputs. (e.g., traffic light, boom gate.) • Understand that a program must be saved for processing at a later stage (Link to D.3, D.4, D.10, C.2, R.6, R.7)	• Present a basic understanding that input results in some form of output. • Illustrate through a basic activity how input results in some form of output. • Understand that different forms of input result in different actions/ outputs. (e.g., traffic light, boom gate.) • Present a basic understanding of the concept that processing takes place between input and output. • Identify output as a form of communication from the device. • Understand that a program must be saved for processing at a later stage. (Link to D.3, D.4, D.10, C.2, R.6, R.7)	• Demonstrate/mimic a simple activity where input results in some form of output. • Distinguish between input through instructions that are executed and results in action and output as a form of communication from the device. • Describe the interaction/relationship between input, processing, and output (e.g. when coding). • An elementary understanding of storage elsewhere (not on device e.g. cloud storage). • Understand that incorrect input results in incorrect output (GIGO) (Link to D.3, D.4, D.10, C.2, R.6, R.7)
D.8 **Interpret a pattern to represent or communicate a message**	• Interpret a foundational pattern (e.g., representations such as a coloured paper or flags or a torch/flashlight) to communicate	• Interpret a basic pattern (e.g., representations such as morse code or a basic cipher) to communicate (decode) a basic message.	• Interpret a simple pattern (e.g., representations such as morse code, binary code, a basic cipher) to communicate (decode) a simple

or image.	(decode) a foundational message to be interpreted/decoded. • Interpret an image (e.g. a 'road sign' or symbolic representations such as smileys) • Decode/decrypt a foundational message. (Link to D.9 and C.1, C.2)	• Interpret an image (e.g. a 'road sign' or symbolic representations such as smileys) • Decode/decrypt a basic message. (Link to D.9 and C.1, C.2, R.5, R.6 and R.7)	message. • Interpret an image (e.g. a 'road sign' or symbolic representations such as smileys) • Decode/decrypt a simple message. (Link to D.9 and C.1, C.2, R.5, R.6 and R.7)
D.9 **Create a pattern to represent or communicate a message or image.**	• Create a foundational pattern to communicate a message (e.g., design a 'road sign' using a grid to communicate a message using pen-and-paper). • Design a message, e.g. a 'road sign' using an XY grid in a block-based application. (Link to D.8 and C.1, C.2)	• Create a basic pattern to communicate a message (e.g., use a cipher such as Caesar cipher to create (encrypt/encode), communicate a 'message or design and communicate a message using symbols such as a 'heart' or smileys using a microcontroller (LEDs on grid). • Simulate/display the message (e.g., symbols such as a smiley or heart) on a microcontroller (LEDs on grid). (Link to D.8, C.1, C.2, R.5, R.6 and R.7)	• Create a simple pattern to communicate a message (e.g., use a simple cipher to create (encode/encrypt) and communicate a message or design an image (e.g., text to communicate a message) • Simulate/display a simple message/ game (e.g., scrolling 'billboard message' or rock, paper, scissors game) on a microcontroller (LEDs on grid). (Link to D.8, C.1, C.2, R.5, R.6 and R.7)
D.10 **Demonstrate a basic proficiency in the application of digital skills.**	• Switch on/power up a computing device, e.g., tablet or PC (hardware). • Open a software application, e.g., block-based coding application. • Work in the IDE of the block-based coding environment and navigate the IDE (software/'app'). • Provide real-world examples, e.g., open and save a program/save a friend's phone number on a mobile phone. • Apply basic file management to open a file (e.g., block-based application) and save a file, e.g., save block-based application file (storage). • Design a simple sprite, using an application such as Paint use in a block-based application (Link to D.9). (Link to C.2 – C.5 and R.5 – R.7)	• Switch on/power up a computing device, e.g., tablet or PC. • Open a software application, e.g., block-based coding application. • Describe and demonstrate the concept of saving files using a descriptive filename. • Create and name a folder for saving files created in block-based coding application and write down the file path. • Describe and demonstrate the concept of opening a file from within an application (e.g., block-based coding application) as well as using a file path (from folder created). • Save and Open filles from within an application as well as following a file path. • Design a simple sprite and a simple backdrop, using an application such as Paint, to use in a block-based application (link to D.9). (Link to C.2 – C.5 and R.5 – R.7)	• Load/open, save, and run a block-based coding application. • Explain and demonstrate the concept of saving files using a descriptive filename and file extension. • Explain the purpose of a file extension. • Create and name a simple folder structure for saving files. • Explain file and storage management – basic file management. • Save and Open filles from within an application as well as following a file path. • Fluent use of different input and output devices to perform tasks and functions. • Design a simple sprite and a simple backdrop to import and use in a block-based application, using an application such as Paint (link to D.9). • Design a customised 'GUI' for a block-based application. (Link to C.2 – C.5 and R.5 – R.7)

Note
Linked competencies can be grouped/done together within one lesson/activity where appropriate.

2.13 Envisaged Learner

The Coding and Robotics learner shows an interest in technology and its application in the world. The learner can think logically and critically and is able to solve problems. Furthermore, the learner is creative and innovative as well as disciplined, focused, and persistent. The learner can also work well with others to achieve a common goal.

2.14 Career Opportunities

Today, digital technologies are integrated in all aspects of our lives. Digital competencies such as Coding and Robotics skills make one more employable and effective in any job and support further studies.

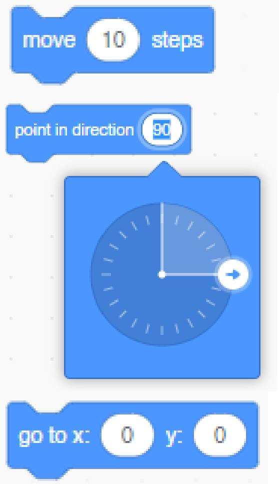
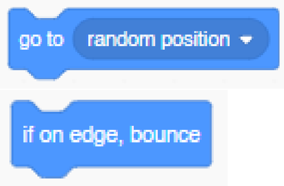
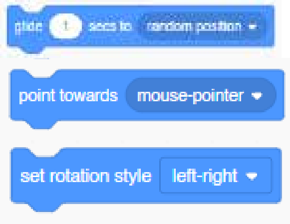
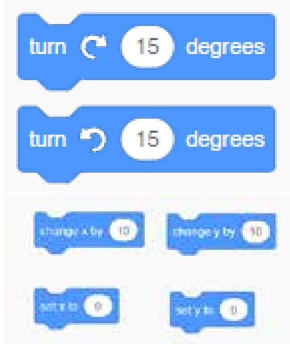
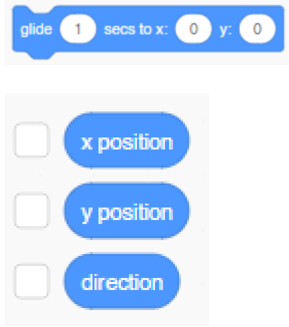
The growing ubiquity of digital technologies and the developments around the Internet of Things (IoT), automation and artificial intelligence (AI) have seen the inclusion of skills such a computational thinking, design thinking, software development (coding) and robotics in every sector of employment and entrepreneurship. Therefore, Coding and Robotics aims to equip learners with knowledge and skills that will allow them to thrive in any career and specifically in careers such as software development, robotics engineering, artificial intelligence, etc.

2.15 Progression and Exit Skills Per Grade of Focus Areas

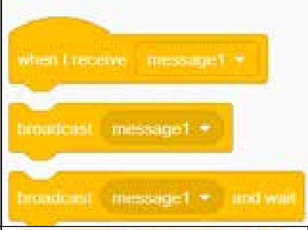
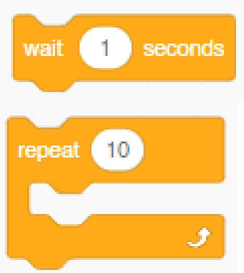
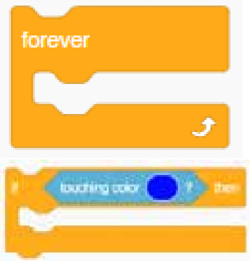
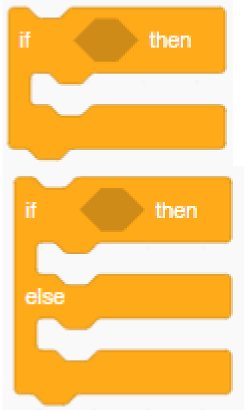
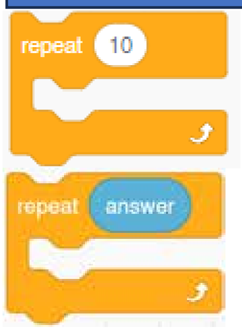
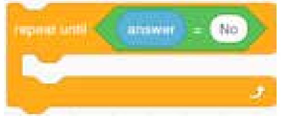
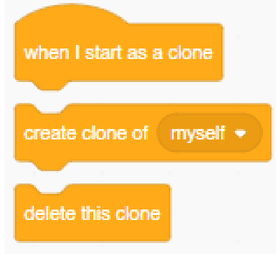
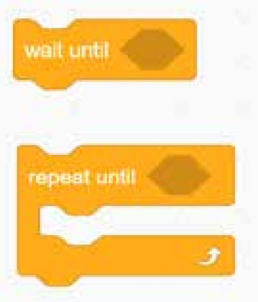
2.15.1 Coding

The following table provides the coding competencies that learners must demonstrate by the end of each Grade in Intermediate Phase:

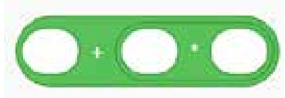
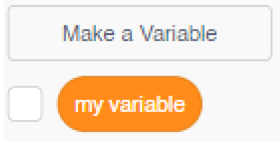
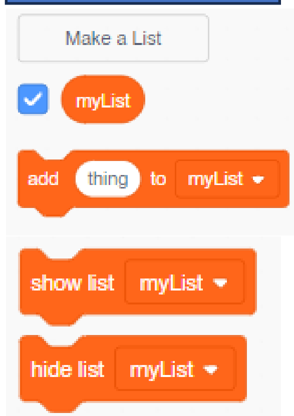
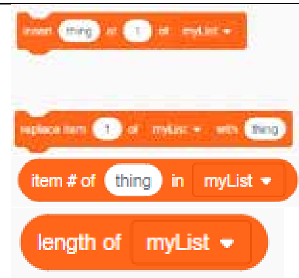
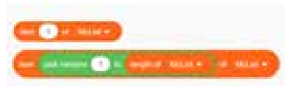
Table 2-7 Intermediate phase coding concepts, content and skills breakdown and progression

	Scratch for Intermediate and Senior phase (content breakdown and concept progression)					
	Grade 4	Grade 5	Grade 6	Grade 7	Grade 8	Grade 9
	 Only 0,0 for middle and to fixed given starting coordinates. 					

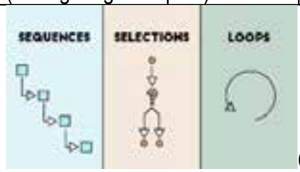
Looks	say Hello! for 2 seconds say Hello! think Hmm... for 2 seconds think Hmm... show hide next costume	switch costume to costume2 change size by 10 set size to 100 % switch backdrop to backdrop1 next backdrop	set color effect to 0 change color effect by 25 color fisheye whirl pixelate mosaic brightness ghost	clear graphic effects costume number backdrop number size	go to front layer go forward 1 layers	
Sound	play sound Meow until done Meow record... change volume by -10	start sound Meow stop all sounds	change pitch effect by 10 set pitch effect to 100 clear sound effects set volume to 100 % volume			
Events	when [green flag] clicked when space key pressed when this sprite clicked	when backdrop switches to backdrop1	when loudness > 10			

						
Control	 Single command. From term 3 onwards a maximum of two related commands can be grouped. E.g., (Move and Point in direction, Move and stamp, Move and wait) etc. If touching colour to continue Grade 3 knowledge 	 If with answer or (key pressed, e.g., space)  If condition + Relational operators with answer block, e.g.,	 With answer as a left side value 	  Fixed counter loop with a single If...then...else allowed	  	  

	stop all Single if with answer block as extended opportunity for highflyers (Optional). if then If with answer	answer > 50 answer < 50 answer = 50			Double nested loop with guidance in problem statement where one loop is a fixed counter loop. NO additional nesting conditions. when I start as a clone create clone of myself delete this clone	Programs should not require more than two nested structures Double nested loops allowed with guidance in the problem statement.
Sensing	ask What's your name? answer	touching mouse-pointer color is touching ? touching color ? mouse down? key space pressed?	distance to mouse-pointer loudness timer reset timer	mouse x mouse y backdrop # of Stage	current year days since 2000	
Operators	+ - * / Fixed values e.g., 5 + 6 join Hello answer	pick random 1 to 10 > 50 < 50 = 50 length of apple	not mod round Stacking of blocks (basic concept mastered in Grade 5) - To nest commands.	letter 1 of apple apple contains a ? sqrt of	floor of abs of 10 ^ of and or	

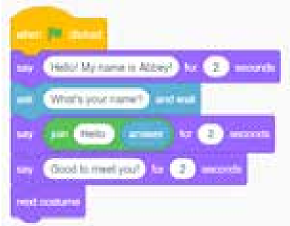
	 Space manually typed after first word.	Stacking of blocks (Nesting commands) 				
 Variables			 Term 2 onwards (Only) Only two variables per application (basic) with guidance 	Limited to three – four variables in the same problem with some guidance in the problem statement	Multiple variables may be introduced with some guidance in the problem statement	
Lists				Create a simple list. Guide the learners with instructions towards its implementation in the solution 	 ** Display random ITEM from a list 	

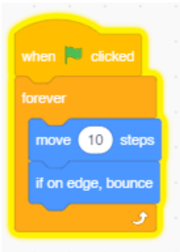
				delete all of myList myList contains thing ?		
User defined Blocks						define MyOwnBlock MyOwnBlock
Pen	erase all stamp pen down pen up set pen color to	change pen color by 10 set pen color to 50 change pen size by 1 set pen size to 1				
Text to speech		set voice to alto speak hello These blocks and functions can be used to illustrate the concept of AI to the learners				
Video sensing			Video Sensing – Optional for highflyer			

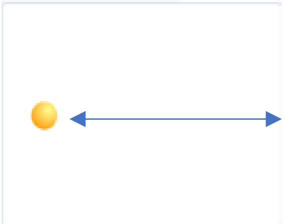
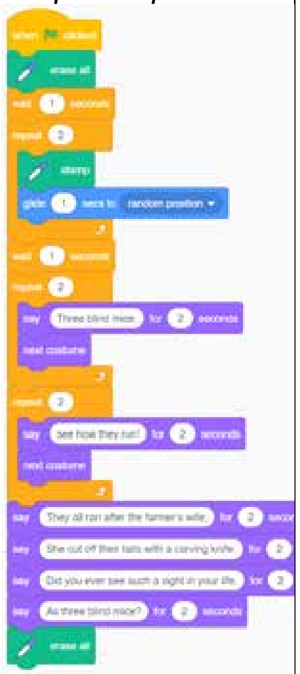
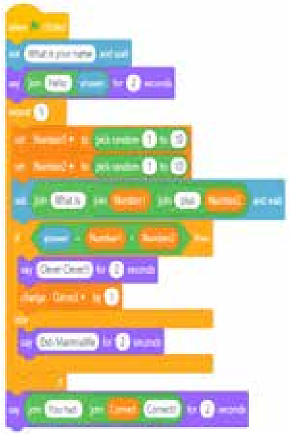
						
Application skills (IDE)	Select your own sprite Select your own background Save your program Open an existing program (Change a given sprite)	Import a picture as a sprite Import an animated gif as a sprite with costumes Import a background				
		**Concepts, constructs and practices**				
	▪ Simple sequential algorithms ▪ Everyday scenarios ▪ Sequences for integration with other subjects (e.g., Languages) + Songs ▪ Iteration on one single command for a fixed number of times ▪ Singular condition (with answer (input value) as reference comparison ▪ Fixed value calculations	▪ Simple sequential algorithms ▪ Everyday scenarios ▪ Sequences for integration with other subjects (e.g., Languages) + Songs ▪ Change the costume of a sprite ▪ Forever loops ▪ Singular condition with else (with answer (input value) as reference comparison ▪ Forever loop + 1 (nested conditional structure)	▪ Simple sequential algorithms ▪ Everyday scenarios ▪ Sequences for integration with other subjects (e.g. Languages) + Songs + Modelling a traffic light. ▪ Change the costume of a sprite ▪ Use of iteration (simple) with variable condition ▪ Change the backdrop ▪ Draw shapes with loops ▪ Forever loops ▪ Loop + Singular nested conditional structure ▪ Stacking (nesting of blocks) for calculations string output. ▪ Guide the learners on the use and implementation of variables	▪ Sequential algorithms ▪ Everyday scenarios with simple problems ▪ Sequences for integration with other subjects (e.g., Languages) + Songs + Technology + Mathematics, Natural Sciences, Life orientation. ▪ Change the costume of a sprite based on a condition or broadcast ▪ Use of iteration (simple) with variable condition ▪ Fixed counter outer loop, with nested simple conditions ▪ Change the backdrop ▪ Draw shapes with loops ▪ Forever loops ▪ Loop + Singular nested conditional structure ▪ Stacking (nesting of blocks) for calculations string output. ▪ Guide the learners on the	▪ Sequential algorithms ▪ Everyday scenarios with simple problems and integration with other subjects with strengthening of concepts using other subject domains. E.g. Smart plant watering system. ▪ Double nested loops with guidance ▪ Sequences for integration with other subjects (e.g. Languages) + Songs + Technology + Mathematics, Natural Sciences, Life orientation. ▪ Change the costume of a sprite based on a condition or broadcast. ▪ Add multiple sprites to a solution including stamping images. ▪ Use of iteration (simple) with variable condition	▪ Sequential algorithms ▪ Everyday scenarios with simple problems and integration with other subjects with strengthening of concepts using other subject domains. E.g., Smart plant watering system. ▪ Double nested loops with guidance ▪ Sequences for integration with other subjects (e.g., Languages) + Songs + Technology + Mathematics, Natural Sciences, Life orientation. ▪ Change the costume of a sprite. ▪ Add multiple sprites to a solution including stamping images. ▪ Use of iteration (simple) with variable condition ▪ Fixed counter outer loop, with

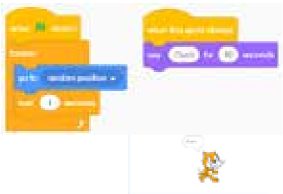
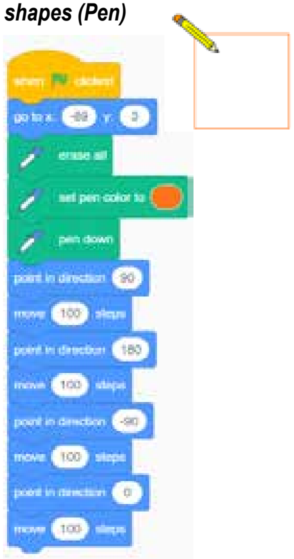
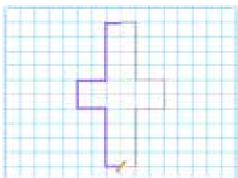
				use and implementation of variables. ▪ Introduce the basic concept of a list (A list should be given as part of a partial or incomplete solution) ▪ Select and display random items from a list (Concept of a list to store and use items, e.g., Values for display)	▪ Fixed counter outer loop, with nested simple conditions ▪ Change the backdrop based on a condition or broadcast. ▪ Draw more integrate shapes with loops based on user input. ▪ Forever loops ▪ Loop + Singular nested conditional structure ▪ Stacking (nesting of blocks) for calculations string output. ▪ Guide the learners on the use and implementation of variables and the development of more complex solutions. ▪ Select and display random items from a list (Concept of a list to store and use items, e.g., Values for display) ▪ Create a simple list	nested simple conditions ▪ Change the backdrop based on a condition or broadcast. ▪ Draw shapes with loops based on user input. ▪ Forever loops ▪ Loop + Singular nested conditional structure ▪ Stacking (nesting of blocks) for calculations string output. ▪ Guide the learners on the use and implementation of variables. ▪ Perform basic operations on lists. ▪ Introduce the concept of procedures through the implementation of lists. ▪ Access and modify an individual element in a list. ▪ Add and delete elements in a list. ▪ Solve more complex problems with guidance in the problem statement.
Use of sprites	One sprite only.	A maximum of two sprites.	A maximum of two sprites.	Maximum of three sprites	As required by the problem	As required by the problem
	Example - Simple sequential block 	***Example with IF answer block*** 	**Example with a loop and singular nested conditional structure** NOTE: In this activity the learners should be instructed and guided towards the use of the variables. The example below can be presented using scaffolding. - Program with two fixed numbers and the answer is checked. - Program with two variables and the answer is checked. - Program including a fixed loop for (e.g., 5 questions) - Adding a variable to count the correct answers			

<table>
<tr>
<td></td>
<td>

Example – Sequential (Input Output)


Example - If on Edge Bounce


</td>
<td>
Example with repeats

</td>
<td>
Learners are guided with the use of variables.</td>
<td></td>
<td></td>
<td></td>
</tr>
</table>

	Example – Fly around (Two events click)  ***Example – Draw simple shapes (Pen)***  ***Shapes pen – Symmetry*** Learners are provided with a stage, with a partial geometrical shape. The learners must then code how to draw the corresponding symmetrical shape. 					

In terms of coding, typically, problems could require learners to

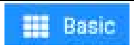
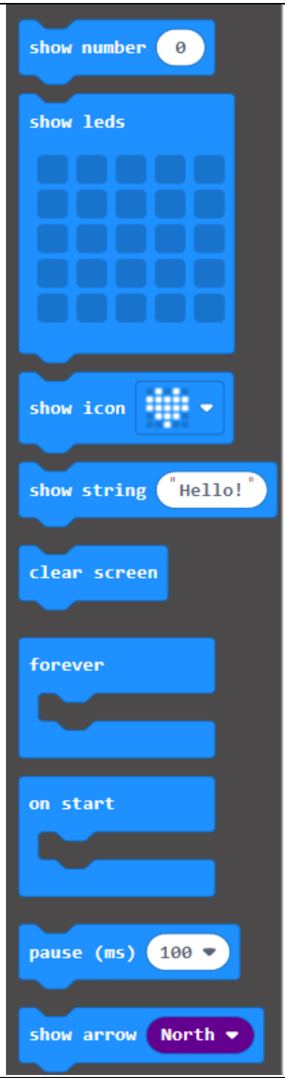
- read code and explain what it does or
- work through (trace) / act out code (physically or simulated) to determine the output or the correctness or
- provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or
- translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions))
- add some functionality/instructions to an existing program.
- rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or
- choose the correct solution from 2-3 options or
- compare different solutions to evaluate efficiency or
- debug an algorithm or block-based program (find the bug, describe the bug and correct it)
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

depending on the competency/(ies) the learner needs to demonstrate.

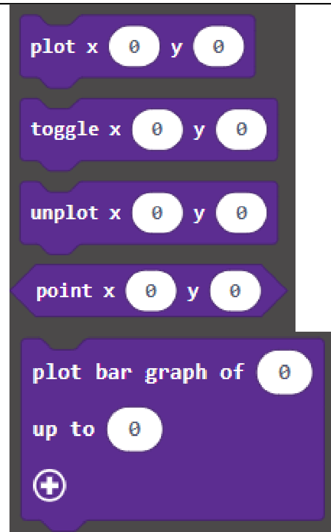
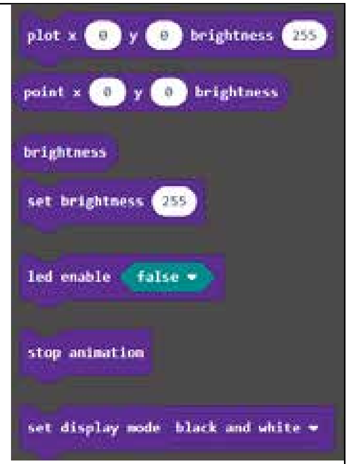
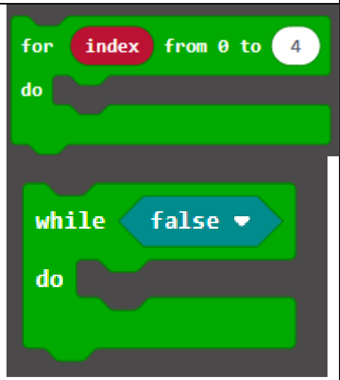
2.15.2 Robotics

The following table provides the robotics competencies that learners must demonstrate by the end of each Grade in Intermediate Phase:

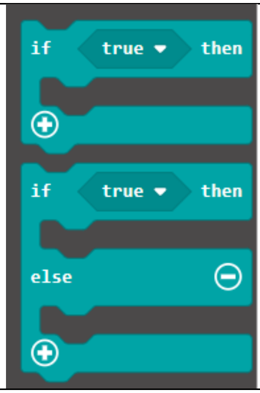
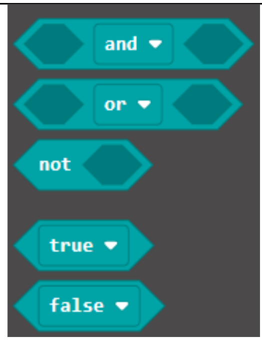
Table 2-8 Intermediate phase robotics concepts, content and skills breakdown and progression

	Make Code (Microbit) for Intermediate and Senior phase (content breakdown and concept progression)					
	Grade 4	Grade 5	Grade 6	Grade 7	Grade 8	Grade 9
						

Input		on button A pressed on shake Shake = sensors describe button A is pressed Clearly differentiate between the on (event) and the if as a conditional construct.	light level compass heading (°) temperature (°C) pin P0 is pressed on pin P0 pressed on pin P0 released	is shake gesture on loud sound on logo pressed logo is pressed sound level	set loud sound threshold to 128 calibrate compass	acceleration (mg) x set accelerometer range 1g
Music		Melody play melody at tempo Tone play tone Middle C for 1 beat rest(ms) 1 beat set volume 127 volume stop all sounds	change tempo by (bpm) 20 set tempo to (bpm) 120 1 beat tempo (bpm)	play melody dadadum in background stop melody all music on melody note played set built-in speaker OFF		

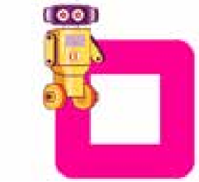
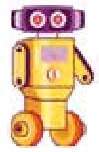
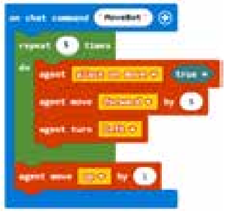
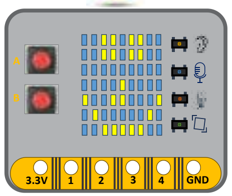
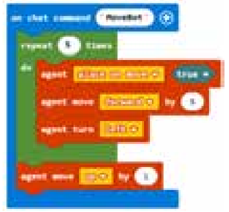
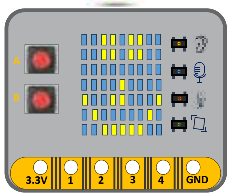
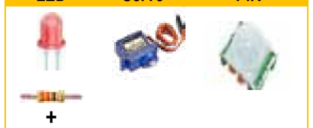
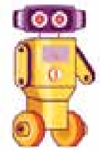
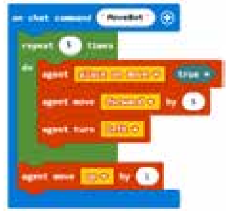
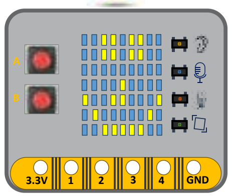
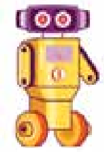
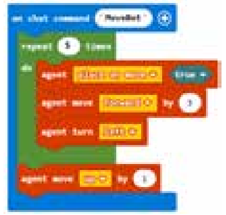
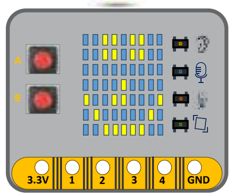
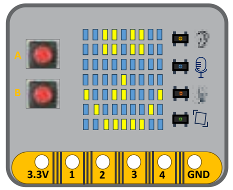
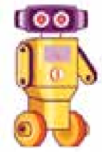
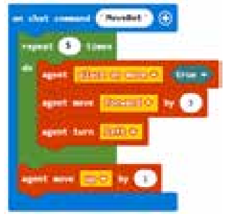
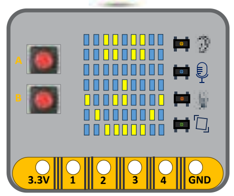
Led						
Loops			 Check while			

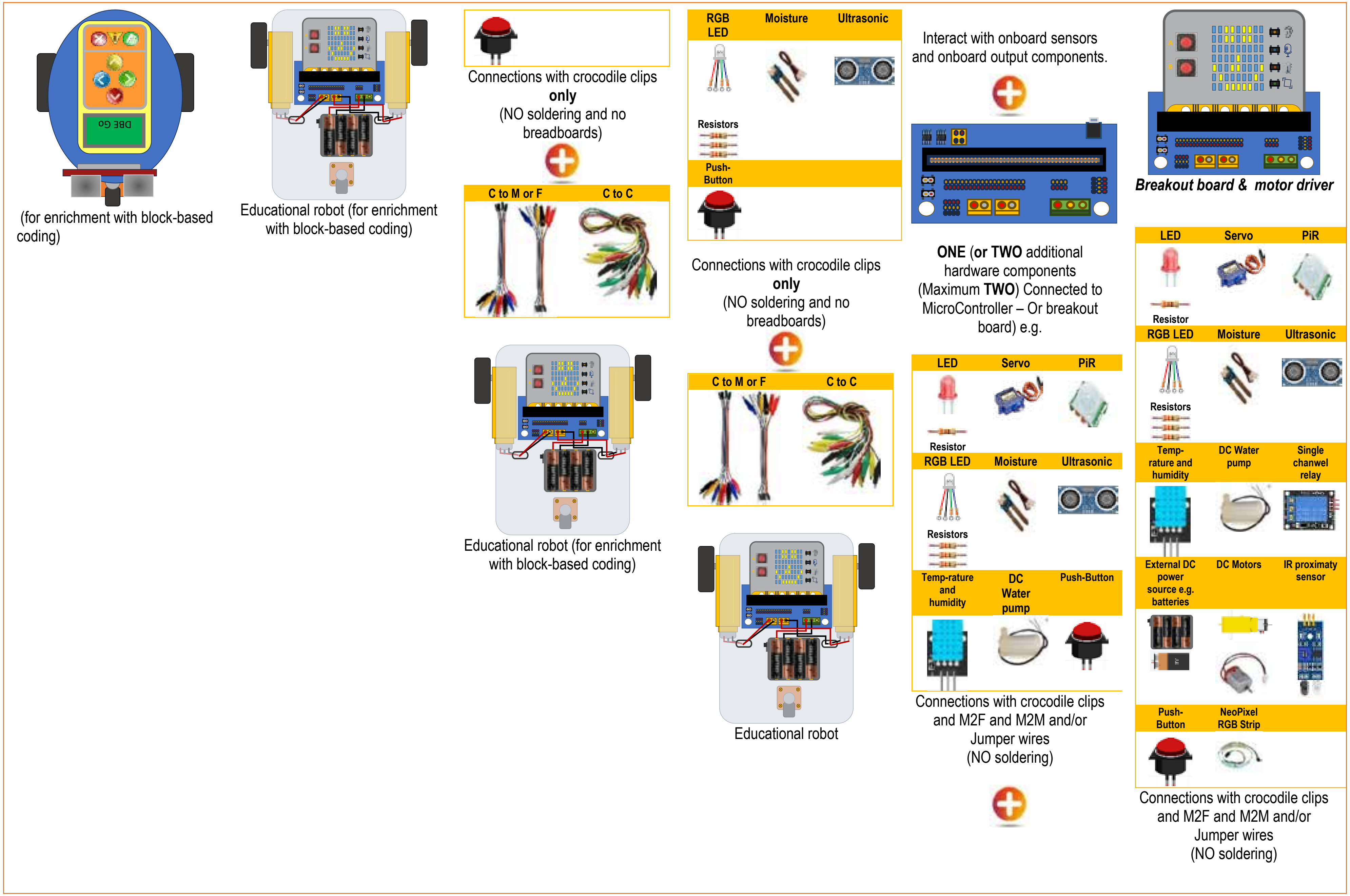
Radio				Group radio set group 1 Send radio send number 0 radio send value "name" = 0 radio send string " " Receive on radio received receivedNumber on radio received name value on radio received receivedString received packet signal strength		

Logic		if true then
		if true then
		else
			and
			or
			not
			true
			false			

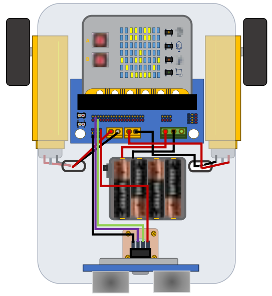
		0 = 0 0 < 0				
Variables			Variables Make a Variable...	Variables Make a Variable... Multiple variables may be introduced with some guidance in the problem statement		
Math		0 + 0 0 - 0 0 x 0 0 ÷ 0 pick random 0 to 10	remainder of 0 ÷ 1 round 0	pick random true or false	remainder of 0 / 1 min of 0 and 0 max of 0 and 0 absolute of 0 square root 0 round 0	
Extenstions						
Application Skills (IDE)		Save your program Open an existing program (Change a given application)	Download a program to the device (Hex file)			

2.15.3 Robotics progression Grade 4 – Grade 9

Grade 4	Grade 5	Grade 6	Grade 7	Grade 8	Grade 9
 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control **Optional Educational Robot –**	 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control  ***Microcontroller*** Interact with onboard sensors and onboard output components only.	 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control  ***Microcontroller*** Interact with onboard sensors and onboard output components. **ONE additional hardware** component e.g. **LED** **Servo** **PiR**  **+** **Resistor** **Push-Button**	 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control  ***Microcontroller*** Interact with onboard sensors and onboard output components. **ONE** (**or TWO** additional hardware components (Maximum)) e.g. **LED** **Servo** **PiR**  **Resistor**	 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control  ONE ***Microcontroller*** (or TWO Microcontrollers for - Radio Communication)  	 ***Virtual robot*** in a code/simulated environment  Block based coding for robot control  ONE ***Microcontroller*** (or TWO Microcontrollers for - Radio Communication) Interact with onboard sensors and onboard output components. 



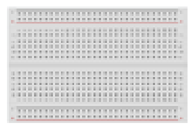
DBE Go
(for enrichment with block-based coding)
Educational robot (for enrichment with block-based coding)
Connections with crocodile clips only (NO soldering and no breadboards)
C to M or F
C to C
Educational robot (for enrichment with block-based coding)
RGB LED
Moisture
Ultrasonic
Resistors
Push-Button
Connections with crocodile clips only (NO soldering and no breadboards)
C to M or F
C to C
Educational robot
Interact with onboard sensors and onboard output components.
ONE (or TWO additional hardware components (Maximum TWO) Connected to MicroController – Or breakout board) e.g.
LED
Servo
PiR
Resistor
RGB LED
Moisture
Ultrasonic
Resistors
Temp-rature and humidity
DC Water pump
Push-Button
Connections with crocodile clips and M2F and M2M and/or Jumper wires (NO soldering)
Breakout board & motor driver
LED
Servo
PiR
Resistor
RGB LED
Moisture
Ultrasonic
Resistors
Temp-rature and humidity
DC Water pump
Single chanwel relay
External DC power source e.g. batteries
DC Motors
IR proximaty sensor
Push-Button
NeoPixel RGB Strip
Connections with crocodile clips and M2F and M2M and/or Jumper wires (NO soldering)

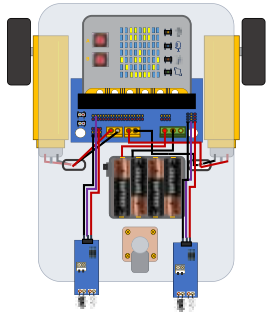


Educational robot + Breakout board & motor driver (For Obstacle avoidance using Ultrasonic sensor only)

NOTE: Learners are not required to assemble a collision avoidance robot. They only need to understand the principles of its operation.

Simple breadboard circuits

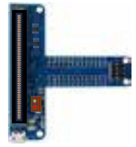
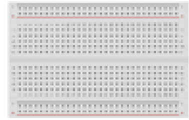
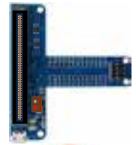




Educational robot + Breakout board & motor driver + Line following robot with 2 IR proximity sensors

NOTE: Learners are not required to assemble a line following robot. They only need to understand the principles of its operation.

Simple breadboard circuits

					  **Optional components – (For enrichment)** **7 Segment Display**  **8x8 Led Display**  **LCD 16x2 Display**   **Resistors** **Additional components for enrichment** - Thermistor
	Simple Sample Projects Based on using board only	**Simple Sample Projects** **(Single component only + Onboard components)** ▪ Soil Moisture with two conductors (e.g. nails) ▪ Monster Munch (Servo project) ▪ Own switch with foil (LED Project) ▪ Simple alarm with PIR and onboard buzzer	**Simple Sample Projects** **(One or two components + Onboard components)** ▪ Automatic dustbin or opener (railway crossing sensor and servo) ▪ Soil Moisture with sensor ▪ RGB LED Project ▪ Ultrasonic sensor Project ▪ Alarm with PIR or Ultrasonic sensor and additional component e.g., External buzzer, LED. ▪ BASIC robot car with 2 x 360 Servo's ▪ Traffic light with 3LED's	**Sample Projects** **(One or two components + Onboard components)** ▪ Obstacle avoidance robot (Only conceptual) ▪ Self-watering plant ▪ "Smart device" (Combination of sensor and responder device)	**Sample Projects** **(Four components Max + Onboard components)** ▪ Line following robot (Only conceptual) ▪ Smart home / Greenhouse (2 Sensors + "2 Actions". E.g., PIR and auto light on & Temperature and auto fan on) ▪ "Smart device" (Combination of sensors and responder device and output display)

3 SECTION 3
CONTENT SPECIFIC CLARIFICATION PER GRADE PER TERM

The following tables provide the content clarification per term and per grade.

This section should be read in conjunction with Tables 2-1 to 2-11 and Figures 2-7 to 2-10.

In Intermediate Phase, the curriculum is designed to also strengthen the specific concepts and content that link to other subjects such as Mathematics, Natural Sciences and Technology and Life Skills.

Content clarification is done with examples as Coding and Robotics is a new subject.

Note:

This section contains examples that clarify the content and competencies. These examples serve as illustrations to better understand the topics and the competencies learners are expected to develop.

However, teachers should see these examples as a starting point for teaching the content and competencies. While the examples are beneficial, teachers should not limit themselves to just those activities. They are encouraged to include other exercises and tasks to ensure deliberate practise, retrieval practice and a deeper understanding of the concepts and skills being taught.

The content and competencies are also grouped based on the main topic areas. This organisation helps teachers understand which skills and knowledge are related and how they are connected. The content and competencies are therefore not necessarily listed in the order they must be taught. Teachers have flexibility in how they sequence the topics based on the context of their teaching environment and the needs of their leaners. However, there is an indication of how different competencies relate to each other. This linkage could help teachers understand the progression of skills and how they support or build upon one another or could be taught in relation with other skills and competencies.

Teachers should therefore develop their Annual Teaching Plans (ATPs) sequencing content and competencies in a manner that will make sense for their learners and their teaching and learning environment to foster a positive learning experience. The goal of developing the ATPs is to maximize the learners' learning outcomes, acquisition of competencies and achievement.

It is also important to note that physical and paper-based activities should not be neglected once learners start to work on a computer.

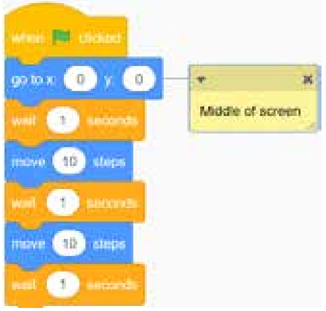
3.1 Grade 4

Note:

Teachers must include the following competencies and content in their Annual Teaching Plan (ATP), distributed across the terms and sequenced, organised and grouped in a manner that will facilitate learning in a manner that will make sense for learning and teaching, maximize the learners' learning outcomes and achievement. and in a way that will make optimal use of time and resources.

3.1.1 Term 1

Content (Grade 4 / Term 1)	**Notes/Examples**
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.2**
Example activity 1 - What is computational thinking? Use the videos (links in notes section) on computational thinking and prepare a worksheet with questions based on the videos for learners to complete afterwards. Learners watch the videos on CT and make notes or write down questions they would like to find answers to using a KWLS chart (Refer to Annexure C). After watching the videos, divide learners into pairs, and hand out a worksheet to each pair to discuss and answer the questions on the worksheet and complete the KWLS chart.  Let some learners report back and discuss the following examples with learners. **Abstraction**: • Your timetable is an example of abstraction of time and activities. It represents a week in terms of days and periods, helping you to prepare for school and to attend the correct class at the correct time. • A plan of the school grounds is an abstraction. It helps you to find the building or classroom that you want to go to. **Decomposition**: • Cleaning your room by first making your bed, then packing away your clothes, then dusting and then vacuuming the floor. • You need to fetch 10 l water from the river to your house in the village. You know that you are not strong enough to carry one container with 10l water. You decide to use a 5l container and doing two trips. **Pattern recognition**: • Noticing that all birds have feathers, two wings, a beak and two legs. • Realising that the difference between terms in a series of even numbers is two, e.g. 10, 12, 14, 16 … **Algorithms**: • Baking a cake following a recipe. The set of rules, steps or instructions to bake a cake is an algorithm. • Directing someone from your home to the nearest shopping centre. • A user manual for assembling something or repairing something.	Learners must understand the following: • What computational thinking is • What abstraction is • What decomposition is • What pattern recognition is • What an algorithm is • What makes a good algorithm • How to use CT to develop a good algorithm that can be coded to implemented in a coding environment Learners must understand that Computational thinking (CT) is an attitude and a skill set where one uses specific techniques and strategies (abstraction, decomposition, pattern recognition, algorithm design) that help one to complete tasks successfully and to solve problems systematically. It further helps us arriving at a solution that both humans and a computer can understand. Encourage learners to become proficient with computational thinking when engaging in all activities in this curriculum. Links to CT videos: https://youtu.be/mUXo-S7gzds (intro) **Note**: These activities would span about 3 – 4 30-minute lessons and are done unplugged (pen-and-paper – no computing device required). **Note**: ***Before proceeding with the activity in C.2, first do activities in D.10*** (switching on computer, open block-based coding application, navigate the coding environment (IDE) and ensure learners are comfortable to implement example activity 4 in the coding environment.

Content (Grade 4 / Term 1)	Notes/Examples
Example activity 2 – What is an algorithm (set of logical instructions) and what makes a good algorithm? Learners write the steps / instructions for making a peanut butter sandwich. Learners then watch the following video: https://www.youtube.com/watch?v=Ct-lOOUqmyY Learners then work in pairs and write down what they think a good algorithm entails and to improve the algorithm for making a peanut butter sandwich. Now that learners have some idea of what CT entails and what makes a good algorithm, proceed with activity 3. **Example activity 3** Divide learners in pairs. Allocate the roles of driver and navigator. Learners use pen-and-paper to create an algorithm to draw a square (do not provide the algorithm) The driver walk-out the square step-by-step and the navigator write down the instructions step-by-step. The pair test the steps by asking and acting it out. **Example** ***Acting out*** Move 10 steps Turn right Move 10 steps Turn right Move 10 steps Turn right Move 10 steps Turn right ***Written*** 1. Turn right. 2. Draw a 3 cm line. 3. Turn right. 4. Draw a 3 cm line. 5. Turn right. 6. Draw a 3 cm line. 7. Turn right. 8. Draw a 3 cm line.	**Note**: A **computer program** is a sequence or set of instructions in a programming language for a computer to follow to perform a specific task. **Note** Evidence suggests that learners should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practise (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to D.10 and D.1** **(Start with D10 before doing C.2)**
Example activity 1 – Introduce Go to middle of screen *Move* and *Wait* Provide learners with the following code on the computer and the following instructions on a worksheet or the board: • Run the code, then inspect the code and explain what the code does. • Add instructions to do the following: • After the sprite moved the first 10 steps and waited 1 second, add an instruction to reduce the size of the sprite by half. • After the sprite moved and waited for the second time, add an instruction for the sprite to think: 'Hmm, I'm getting smaller as I move forward' for 2 seconds. • Run the changed code and ensure that it works.  **Example activity 2 – Introduce Pen extension with draw and Turn Left/Right 90 degrees** (proceed from C.1 activity 4) Then learners work in pairs to translate the algorithm done in activity 4 (C.1) into block-based code and run the program in a block-based coding environment. One learner fulfils the role of "driver" and the other "navigator". If working with computer/device the driver is the one managing the device and typing. The navigator takes direction from the teacher and consults the teacher and resources. Learners will need to add the pen extension and change the default sprite to a pen (reducing the size of the pen is also advisable) The activity requires the drawing tool "turn" several degrees (they need to figure out that 'turn right' (in the algorithm) means 90 degrees)) and moving forward several pixels/steps. Blocks to be introduced with this activity: • When green flag is clicked (event) • Go to a specific position on the backdrop/grid (learner decide)	**Leaners will first be introduced to the block-based coding platform (Refer to D.10) – use** Tutorials For activity 1, guide them on how to add the drawing tools (pen) Click on [icon] in the left-hand corner of the IDE – a window will open and click on Pen. The pen extension is added and a Pen in the ***blocks*** pallet on the left-hand side. Also guide them to change the default sprite to a pen or let them look at the tutorial. Introduce the code blocks as they are used in the activities. **Refer to Table 2-7.** Initially, focus on sequential coding – having instructions in the correct sequence is important in coding and learners sometimes struggle with this. **Sequencing** is putting events or information in a specific order. It is the skill that to plan what steps to take in which order to perform a task successfully. When sequencing, we learn about patterns in relationships, and we learn to understand the order of things. It also helps to develop the

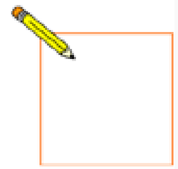
Content (Grade 4 / Term 1)

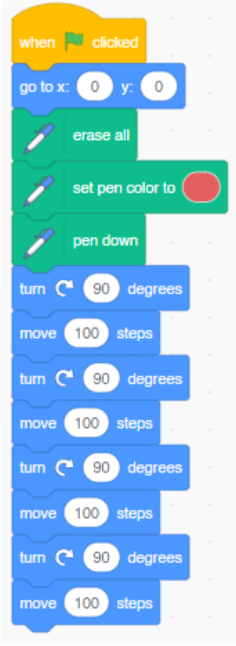
- Direction in which sprite/object must 'face'
- Basic pen/draw commands

Other skills to be introduced:

- Change sprite/object
- Change size of sprite/object
- Change pen colour
- Optional: change line thickness (pen size)

1. Turn right.
2. Draw a 3 cm line.
3. Turn right.
4. Draw a 3 cm line.
5. Turn right.
6. Draw a 3 cm line.
7. Turn right.
8. Draw a 3 cm line.





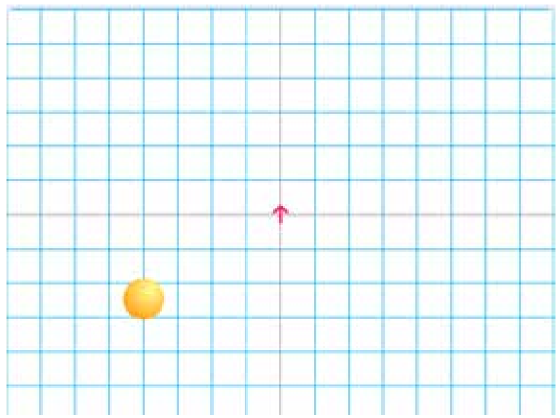
C_4_1_Blokkie.sb3

Note: Allow learners to tinker and figure this out themselves. Do not merely tell them how to solve it. If learners struggle and the navigator requests help/resources, teachers can provide a video for them to watch, e.g. https://youtu.be/-kKMV-iCpy0

Example Activity 3 – Follow instructions – Introduce Point in direction

Use the following instructions (algorithm) to code a block-based program:
Open the block-based application Grid1.sb.:
Change the Cat sprite to an arrow and set its size to 50
Change the size of the ball sprite to 80 (the background should now look like the one on the right)
Write code for the arrow to do the following

1. Go to position (0;0) on the grid
2. Point in direction 180
3. Move 120 steps
4. Turn right
5. Move 90 steps (onto the ball)
6. Turn around (point in opposite direction)
7. Now, move (walk) back to original position (0;0) by reversing steps 3 to 5 (do not use the goto instruction)



Example activity 4 – follow instructions – Change background and costume

Open the block-based application
Change the background to 30 px grid (1 block = 30 steps)
Change the sprite to the balloon
Write code to move the sprite 5 blocks west (to the left) from its current position.
Change to the next costume and wait 2 seconds
Change to the next costume again and wait 2 seconds

Example Activity 5 – open ended (Link to C.7)

Learners work individually and use what they have learned so far and create a block-based coding app of their choice. They first need to plan their app and write an algorithm (which they can do at home). Then code the algorithm (in class), execute the code and make ensure it works.

Notes/Examples

ability to understand and arrange purposeful patterns of actions, behaviours, ideas, or thoughts.
Attention to detail is also important as it helps prevent mistakes and ensures successful completion of a task.

Note: Debugging at this stage is therefore incidental learning – it is only addressed formally in Term 2

Saving the program files is done in relation to D.10.
Initially, use default folder.

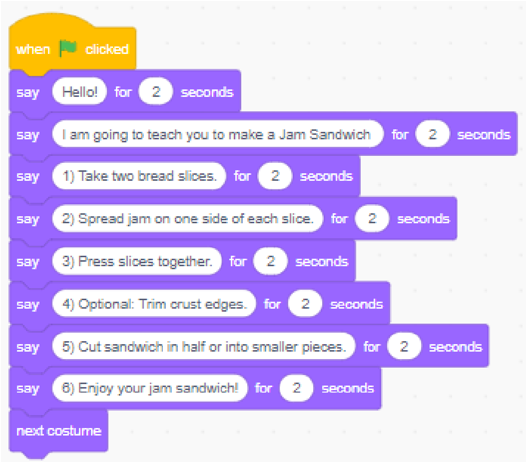
Note:
These activities are based on discovery learning where learners can tinker. (Debugging at this stage is therefore incidental learning – it is only addressed formally in Term 2). Start with activity 1 once done with D.10 (introduction to app

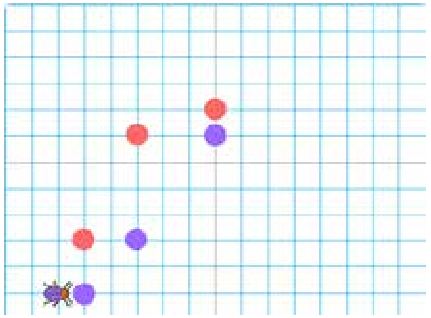
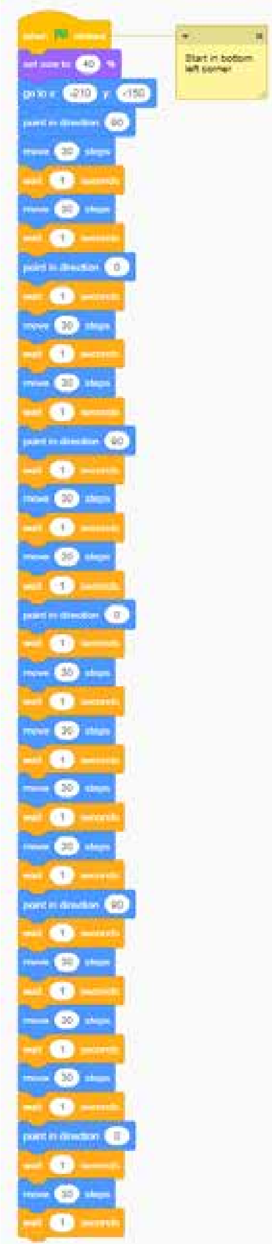
Note
Ensure that learners understand all new blocks/cod/concepts introduced.

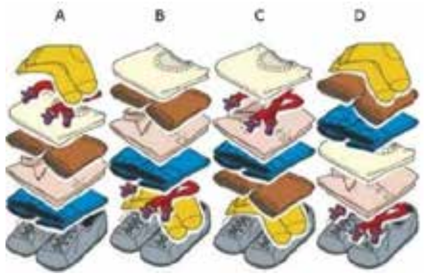
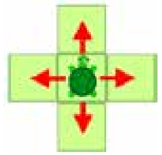
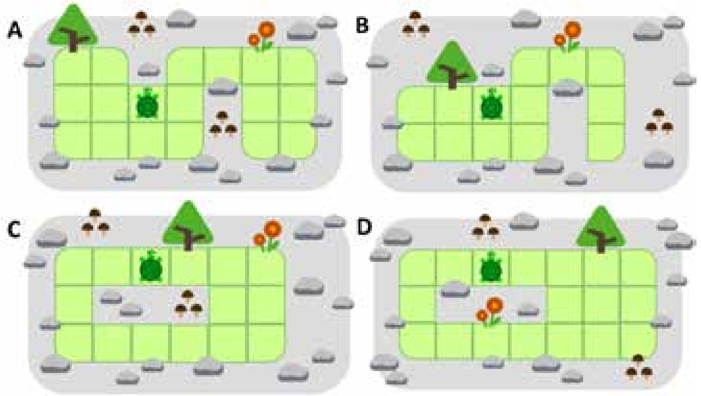
Note:
Activity 3 could be done on a grid outside and learners acting the algorithm out as first step.

Note
Activity 4 can also be linked to C.7

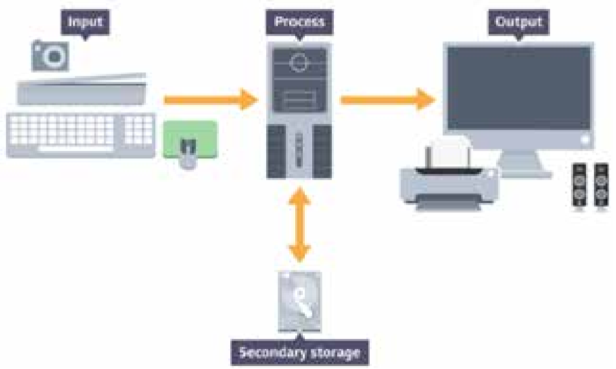
Note:
Ensure that learners understand the basic structure of a block-based program:
Start (when green flag is clicked)
Code blocks (sequence of instructions)
End (stop all)

Content (Grade 4 / Term 1)	Notes/Examples
C.3 Interpret and execute a given symbolic or written set of commands	**Link to C.1, C.2 and R.6 and D.8 -D.10**
Example activity 1 Provide learners with a set of block-based instructions (from the blocks already covered) on a worksheet (unplugged) and ask them to explain what the code does (what will happen if the code is run). Then let them run the code and compare their explanation with what happens. **Example activity 2** Provide learners with an algorithm which they need to must explain regarding what it does/what output it will give, then translate the instructions to code and execute to see if they explained it correctly (program with code blocks already encountered and 1 new block/concept) **Example activity 3** Provide learners with the following program and let them load (from default folder on their computers -refer to D.10 and run the program and inspect the code).  Then ask them to write a similar program (e.g. make a cup of coffee). Learners swap programs and evaluate each other's algorithms (e.g. make a cup of coffee) against their understanding about features that makes a good algorithm (C.1) **Example activity 4 – Introduce *FOREVER* loop and *Next backdrop*** Provide learners with the following code on paper:  C_4_1_Backdrop Change Colour loop_V1.sb3 Blocks introduced: • When key pressed – event that triggers next backdrop to show. • Clear graphics effects – reset the appearance of a sprite/object • Forever loop (instruction inside will keep running as long as program is active) • Play specific sound • They can also watch the tutorial *Add a backdrop* In pairs, learners figure out what the code does. Learners then code the example provided in the block-based environment and run the code, comparing the outcome to their interpretation. Teacher now explains the new blocks/concepts introduced with this activity. Learners are then requested to add an additional functionality (add additional code to do something e) to the program.	Literature suggests that it is important that learners must also read and explain in plain language (their own words) what the code does. This type of activities should be done unplugged (pen-and-paper) and only implemented after learners explained the results. Many of these types of exercises are necessary to ground concepts, skills and understanding of algorithms and coding. **Note:** While learners should be able to describe what each line (block) of code does, (describing a code segment line-by-line/block-by-block) it is very important that learners explain the overall purpose of the code, i.e. what the program does/the purpose of the program is. **Note:** When interpreting the given commands, reiterate the use of decompression and abstraction in the process. **The bug walk activity also links to R.6** (simulate the operations of a virtual robot (the bug can be seen as a virtual robot)) In Grade 4, keep to simple, basic, small activities teaching one or two concepts at a time with lots of repetition to ensure that coding concepts and principles are well grounded and to avoid misconceptions. However also allow learner who are ready to tinker just beyond their comfort zone but avoid giving them tasks that are too complicated as these may impede their self-efficacy. **Note:** Learners need to be exposed to a wide variety of coding problems. Typically, at this stage, problems could require learners to • read code and explain what it does. • work through (trace) / act out code (physically or simulated) / using pen-and-paper to determine the output or the correctness. • provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete. • translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions)). • Add additional functionality to code provided/and existing program

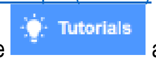
Content (Grade 4 / Term 1)	Notes/Examples
Example activity 5 Provide the following on a worksheet: The stage and the code (on the right) and ask learners to explain what the code does (without running the code – just by reading and interpreting it):  C_4_1_Bug walk.sb3  Discuss their answers, then run the code (for all learners to see) and let them compare their answers to what they see happens when the code is executed and let them reflect on their own interpretations and discuss what they might have interpreted incorrectly and why. Then explain code where necessary. **Example activity 6** When done with activity 5, ask the learners to write an algorithm (using computational thinking) for a similar activity (let their 'robot' move in a specific pattern). They first need to design the activity on a grid, then write down the steps (algorithm). When the algorithm is done, let them translate it into block-based code and implement it in a block-based application. They need to get it to work correctly (though debugging at this stage is therefore incidental learning – it is only addressed formally in Term 2). **Example activity 7** Let learners design another, similar activity using a grid and translate it into code (unplugged). Now, in pairs let learners swap their code (which is on paper) and explain each other's code to each other to see if they can interpret it correctly. Afterwards they can run each other's programs to see if they interpreted it correctly (if the code does not work, it must be corrected – incidentally learning re debugging) **Example activity 8** Open-ended (individual) Use what you have learned so for and write a program of your choice. In groups of 4, let learners demonstrate their programs and discuss them in the groups	• debug an algorithm or block-based program (find the bug, describe the bug and correct it). • develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging. When ***sequencing***, one learns to understand the order of things and about patterns and relationships. By learning to sequence, we develop the ability to understand and arrange purposeful patterns of actions, behaviours, ideas, or thoughts that supports the logical sequencing of coding instructions. In terms of problems that provide a partial solution where some code instructions are missing and learners must fill in the missing code instructions, the concept of Parsons Puzzles could be helpful as it provides scaffolding for learning programming. It helps learners to develop logical thinking, The concept is a type of scaffolded program construction tasks where the learner is given a set of code blocks of a single or multiple lines of code, and the task is to piece together a program from these or to fill in missing code from these. **Example 1 – Fill in missing coding instructions using blocks provided** Provide a problem description and a partial program to solve the problem in the scripts area (leave gaps where missing code instructions should be placed). Also provide the missing code blocks randomly placed (not in sequence) in the scripts area. Learners need to figure out where the missing code blocks fit to complete the program and solve the problem. **Example 2 – Complete a program using code blocks provided** Provide a problem description and all the code blocks to solve the problem, randomly placed in the scripts area (not in sequence) Learners then need to **fit the blocks of code together in the correct sequence** to solve the problem and ensure it function correctly. Parsons programming puzzles are an evidence-based teaching practice that reduces the cognitive load and time spent for learners.

Content (Grade 4 / Term 1)	Notes/Examples
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to C.1, C.2, C.3 and D.8, D.9**
Example activity 1 Bruno has seven kinds of clothes:  *Figure 11 https://olympiad.org.za/talent-search/past-papers/pen-and-paper/*  • Bruno's dad carefully arranges his clothes into four piles • Bruno puts on his clothes in the order that they are in the pile, starting from the top of the pile. • Bruno wants to wear the braces over his shirt. With which piles (A B C D) will Bruno be happy with? **Example activity 2**  Turtles live in small gardens. Each garden is divided into squares, covered with either grass or stones. The turtles cannot cross stony areas. But they can move from one grass square to the next, as shown in the picture. Each turtle needs to take a feeding path in its garden: • It needs to move to all grass squares while visiting each of them only once. • Unfortunately, one turtle cannot take a feeding path in its garden. Which one? Select its garden (A B C or D below.  2022-TS-Elementary-Question-Paper.pdf (olympiad.org.za) **Example activity 3** 1. Look at the code for drawing a block (square) in C.2 Activity 2. 2. Now write an algorithm for drawing a rectangle (use your experience from doing activity 2 in C.2), following a similar pattern Ask yourself questions such as how does a square differ from a rectangle and how are they similar? Use your previous experience about drawing a square. 3. When done, translate the algorithm into block-based code and run the program. Did it work?	Use both pen-and-paper (unplugged) activities and coding activities to expose learners to recognise and interpret patterns. Explain to learners that, one uses computational thinking subconsciously daily and that with computational thinking, one also uses **previous experience** to help one do similar or new tasks or solve similar or new problems, for example, just think about baking cupcakes: One breaks the task of baking cupcakes into smaller tasks such as preheating the oven, mixing the batter and preparing the icing while the cupcakes are baking – one small task at a time. One may also use **previous experience** from baking cupcakes when knowing to bake them slightly longer than the recipe calls for. One also knows that chocolate chips are not a vital ingredient in cupcakes, so one can skip that step if one does not have any available. One also knows to start preheating the oven before pouring the mix into the cups and that, when one takes them out of the oven, one needs to let them cool down before putting on the icing. **As one gets more experienced**, one may also realise that one could prepare the icing the day before. The above will help learners to understand that they can use their experience with activity 2 (C.2) to help them complete activity 3. Learners must be encouraged to use their experience to solve similar or new problems or complete similar or new tasks Pattern recognition is the process to identify and extract meaningful patterns from a dataset. It involves using analysing a set of data to find regularities or repeating structures that can be used to make predictions, classify objects, or solve problems.

Content (Grade 4 / Term 1)	Notes/Examples
Robotics	
R.1 Explain what a robot is in simple terms.	**Link to R.2**
Ask learners what they think a robot is and use their descriptions to formulate a simple definition for 'wat a robot is'. (R,1 and R.2 can be done together)	A robot is a machine that can move independently, sense its environment, make decisions and perform actions. It can be programmed and controlled by humans or operate autonomously. Robots can mostly do the following typical things: sense, compute and act. [What Is a robot? - ROBOTS: Your Guide to the World of Robotics (robotsguide.com)]
R.2 Identify different types of robots.	**Link to R.1**
Explain, at an elementary level, to learners that one gets virtual as well as physical robots: **Virtual robots** These are software programs designed to ***simulate*** the actions and behaviours of physical robots, such as the sprite/object in the block-based programming app. A sprite operates within a virtual environment. When working a block-based environment, one can mimic a robot. **Physical robots** These are tangible, mechanical machines that can interact with the physical world. They have a physical presence and can move, manipulate objects, and interact with their surroundings. They are designed to perform specific tasks in real-word environments, such as robot in a factory that assembles cars. A physical robot operates in the real world. Emphasise that both are programmed and respond to instructions – the one in the real-world environment and the one in a software environment.	**Note:** Can be done with R.1 if time allows Learners need to • acknowledge that robots are diverse and used for different purposes. • point out the similarities and differences between virtual and physical robots. As learners are working in a block-based coding environment, one can use the sprite/object in this environment to discuss the concept of virtual robots (link toC.1-C.7 (virtual robot)) The concept of a sprite in a block-based coding environment shares similarities with a physical robot in that **both are *programmable*** entities that respond to commands and interact with their environment. While a physical robot operates in the real world, a sprite operates within the virtual environment of a block-based project. Users can give commands to the sprite to make it perform certain actions, just like programming a robot to carry out specific tasks
R.6 Mimic the operations of a robot	**Done in relation to C.1 and C.2 and C.3**
Example activity 1 Learners design an activity on a grid and place obstacles and write instructions to move a 'robot' from one point to another, following rules and/or avoiding obstacles (like C.6 activity 2), then act out the instructions (learner acts as robot and follow instructions). **Example activity 2** Learners now code the activity designed in activity 1 using a block-based coding environment (like C.3 activity 5)	Refer to grid activities in C.3 and C.6 and C.7 Activities can be done unplugged (e.g. physical grid (like foundation phase – acting out instructions) on floor or with pen-and-paper) as a transition from foundation phase.
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.3**
<u>Example activity:</u> Classification of technological and non-technological artefacts Using pictures and artefacts of technology and non-technology, learners classify these into technology or non-technology and identify the purpose of each. They must also group the pictures and artefacts as technology only and information technology and describe the difference. The activity/discussion must help them to, at an elementary level, explain what technology is, what information technology is (and by implication the difference and purpose of each) and provide examples of both. This can be done as cooperative learning in groups.	Technology is a broad term for using tools, machines, techniques and processes with the purpose to accomplish a task or solve a problem. Examples are inventions such as the wheel, electricity, computers and mobile phones. Information technology is a subset of technology that focuses on the use of computers as well as hardware and software with the purpose to store, process, retrieve and manage data and information and includes various computing devices.
D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.1 and D.3**
Discuss the concepts of digital world and digital citizenship.	**(D.2 and D.3 can be done together)**

Content (Grade 4 / Term 1)	Notes/Examples
As citizens we use computing devices – ask learners what they use computing devices for. (Now link to D.3 to learn about the concept of a computing device and how to care for a computing device) **Taking care of devices**: General care issues such as cleanliness: Keep hands clean and avoid eating or drinking near the device. Screen care – cleaning gently with a microfiber cloth. Place the device in a protective case when carrying it around to prevent accidental drops or scratches. Properly shut down the device. Immediately inform the teacher if you notice any problems with the device, such as malfunctioning keys loose connections, or unexpected behaviour.	Learners need to acknowledge that the digital world is a virtual/online environment and is created using digital technologies. It forms part of a huge, interconnected network that allows us to use devices to communicate, share information and interact with each other in various ways. We need to use these technologies responsibly and take care of the devices we use in the classroom. Digital citizenship refers to the responsible, ethical, and safe use of digital technology and the internet. Li As digital citizens we need to act responsibly and respectful.
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.1, D.2 and D.7 and C.2 and C.3**
 The teacher briefly revises the concept of what technological artefacts are, what their purpose is, and that technology comprises different components. All general-purpose computing devices, generally follow the same basic model of input, processing, and output: Generally, in elementary terms, the computing device receives input though an input device, processes input received and provide the result as output through an output device. The computing device generally also store data. General purpose computers - Computers - Edexcel - GCSE Computer Science Revision - Edexcel - BBC Bitesize **Example activity:** Caring for my device. Focus on the use of the devices at home and the ones they use in the classroom. In pairs, provide learners with a random list of instructions to care for e.g., pets/devices/something precious. Learners need to indicate in a second column of the list if the example is related to a technological device and provide feedback on certain aspects on the list. Based on the feedback, teacher addresses misconceptions and discusses caring about the devices in the classroom. Consolidate by letting each learner create his/her own list of how to care for the devices and discuss and exchange ideas. Then finalise a list for the class on how to care for the devices in the classroom/computer lab. Every time learners enter the class, remind them of the rules they created for caring about the classroom devices and that as digital citizens they need to be responsible and respect the devices.	A computing device is an electronic device that can process data, perform calculations, and execute tasks based on instructions provided by the user or pre-programmed software. They can take input, process the input (data) and then provide output. Computing devices are designed to perform various operations and solve problems quickly and efficiently.
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.3 and C.1, C.2 and C.3**
Introduce the concept of input, processing, and output by explaining that these are the three main steps that occur in a computer or any other device to perform tasks. **Example activity:** Magical Kitchen – Learners then act out the concept of input, processing, output as follows: Draw three large, labelled columns on the poster board or whiteboard: "Input," "Processing," and "Output." In the activity, learners will be running a magical kitchen, where they must prepare delicious meals using various ingredients. Hand out small pieces of paper to each learner and ask them to write down or draw a food item they would like to cook in the magical kitchen. Once everyone has written or drawn their food item, ask the learners to gather around the "Input" column on the poster board or whiteboard. Then ask the learners to share their chosen food item and place it in the "Input" column. Explain that this represents the input stage where the ingredients or information is entered into the magical kitchen. After all the food items are placed in the "Input" column, move to the "Processing" column. Explain that this is where the magical kitchen processes the ingredients to create a delicious meal. Take the food items from the "Input" column and pretend to mix, chop, or cook them in a magical way. Once the processing is complete, move to the "Output" column. Explain that this is where the magical kitchen presents the result.	Input, output and processing are the three main steps that occur in a computer or any other computing devices to perform tasks. All general-purpose computing devices follow the same basic model: (link to D.3) Input is from an input device such as a keyboard, mouse, camera or touch screen. The processor (CPU) receives instructions and data from an input or storage device. The instructions and data are processed by the CPU and the results are either sent to an output device such as the monitor or speaker or transferred to a storage device. When learners create or execute their programs using the block-based programming environment, reiterate this process by referring the program that receives input (click green flag), process/follow the

Content (Grade 4 / Term 1)	Notes/Examples
Ask the learners to draw what the finished dish would look like. Then, place the imaginary finished dishes or pictures representing the cooked meals in the "Output" column. Introduce the concept of "Storage" by asking where the food/leftovers will be kept. Briefly reiterate the concepts of input, processing, output and storage by referring to the model in D.3 as well as a computer, tablet or mobile phone. Discuss how the 'magic kitchen' activity links to the input, processing and output when working with a computer.	instructions (code) and provide output (actions) based on the processing (executing the code)
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.2 and C.3**

When working on the computer in the block-based coding environment, guide learners to become familiar with the environment by being able to switch on a device, open and close applications.
This must be done in relation to C.2. (When learners start using the block-based coding environment)
Teacher explicitly guides learners through the process of switching on, opening the block-based coding application and to understand that they work in an integrated development environment (IDE)
Teacher shows them.
• the main parts of the IDE and explain what it is.
    o IDE – where one develops programs.
    o Stage/backdrop – where your project comes alive (one sees the results of one's coding) The stage could also have a backdrop (background for your code/story
    o Sprites – objects or 'characters' that appear on the screen.
    o Script/code area – the collections of blocks that are interlocked. These blocks determine how sprites react on the stage.
    o Blocks palette – the blocks one uses to create code (the coloured dot indicates the type of blocks) and the blocks displayed will depend on the type.
• basic file management
    o Open a block-based coding file from the default location.
    o Save a block-based coding file in the default location.
• how to navigate the environment and let them start working through the tutorials at the top of the screen.
	Show learners the block-based coding environment and explain what each section means or let them watch a video:
	https://youtu.be/NqMd44Oi2l4 (Intro to coding environment)
	Show them the sections for the programming palette, where they code (scripts section), the stage, etc.
	Getting-Started-With-Scratch-3.0.pdf (mit.edu)
	Let learners work through the  at the top of the screen. Start with ***Getting Started*** tutorial to introduce the green flag event and adding a backdrop.
	Show and explain on a just-in-time basis – what they will need at a particular stage or for completing a specific activity.
	The process of input, processing/storage and output can also be demonstrated by saving a friend's phone number on a mobile phone.
	Teachers could also let learners watch videos introducing the environment such as https://youtu.be/-kKMV-iCpy0
	
	Note: There is a wealth of material, including tutorials and videos that explains the block-based environment

3.1.2 Term 2

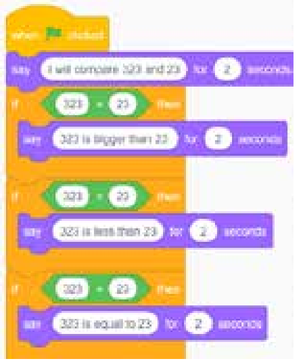
Content (Grade 4 / Term 2)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.2-C.7, R.5-R.7, D.8-D.9**
Example activity 1 Building a tower using Lego blocks Revise, from term 1, what computational thinking is and what an algorithm is. Explain that they are going to focus on characteristics of a good algorithm. Divide learners into small groups (not more than 4) and provide each group with a set of blocks/bricks (e.g. Lego bricks) and an activity worksheet that contains a sequence of steps to follow for building a tower. The instructions must include a mixture of specific instructions and some missing details or mistakes in the sequence. The learners must fill in the missing details or correct the errors to build the tower correctly. (missing details could be the exact placement and orientation of bricks, lack of detail such as size or colour of blocks, and challenges could be incorrect sequence of steps, more than one way to interpret an instruction (ambiguity) etc.) Learners must follow the instructions literally as presented and realise that it does not result in the correct end-product. Example of possible instructions: 1. Place a 2x2 brick at the bottom 2. Add a yellow brick on to 3. Add two a 2x4 brick on top of the yellow brick 4. Add another brick on top of the previous one 5. Finally add a 2x6 brick on the top Groups present their final products and compare with that of other groups and then answer the following questions: Groups then answer the following questions: Why is it essential to follow the instructions in the correct order? How did you handle the missing details or errors in the algorithm? What happens when steps are missed or out of order? Groups now swop instructions and must improve the instructions received by filling in missing details, correcting the sequence, etc. (focusing on detail and sequence) **Example activity 2** You need to explain to someone that is using WhatsApp for the first time how to send a WhatsApp message. You found the following instructions to send a WhatsApp message: • Type message • Open WhatsApp • Send message Rewrite the above instructions to include more detail/steps to make them more precise so that anyone that follows the steps will exactly know what to do and be able to perform the task successfully. Then hand your instructions to a friend to check your instructions for sequence and detail.	Both **sequence** and **detail** are important when developing an algorithm Attention to **detail** is also important as it helps prevent mistakes and ensures successful completion of a task. ***Detail*** means considering every aspect or minor part of something. It is to describe or give ***exact*** information about something. The steps or instructions to perform a task need to be ***unambiguous*** – they need to be ***precise*** and clear to avoid misinterpretation or different interpretations by different people. **An Algorithm** is a set of well-defined steps or instructions that are followed to perform a specific task or solve a particular problem. The instruction set can be sequential or can include branching (decision structure) or repetition (loops). **Key characteristics of a good algorithm: Each step** • must be clear and unambiguous. • must be at the right level of ***detail*** and specific. • consists of a single task (be at the most basic level) • must be in the correct, logical ***sequence*** • must be correct/solve the problem **Remember** One uses CT in all tasks that one wants to complete appropriately, as it helps one to approach problems more systematically and develop well-structured solutions. or find an efficient an effective solution for
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C.1, C.3 and C.4**
Example activity 1 – Introduce *if on edge, bounce* Provide learners with a worksheet with the following code (on the right). First, they need to explain what the code does, then run the code to see if they could describe the function of the code correctly. Now, ensure that learners understand the instruction, *if on edge bounce* **C42SingleSpriteForever**  **Example Activity 2 – Random position and next costume** The CAT sprite has two costumes,  and.  If it switches between the two costumes, it looks like it is walking.	Literature suggests that learners need to read and explain code before they write code.

Write code to do the following: The cat must:

1. Go to the middle of the screen.
2. Switch to costume 1
3. Walk 10 steps
4. Change to the next costume
5. Wait 1 second
6. Add 3 ,4 and 5 another 7 times below.
7. Run the program

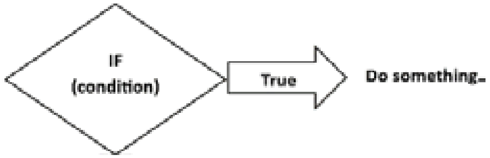
Activity 3 – Random position
Provide learners with the code on the right. Let them first read the code and explain what it does/predict what it would do, then execute the code and compare the results with what they predicted.

Example activity 4 – Introduce IF…THEN
Provide learners with the following code (on the right)
Let learners study the code and explain what it does and what the output will be.
Let them run the code and compare with their explanation and output prediction



Now, explain to learners how the IF-THEN statement works, showing them the following:

Example activity 5 – IF..THEN with ANSWER block
Provide learners with the code on the right
Let learners study the code and explain what it does and what the output will be.
Let them run the code and compare with their explanation and output prediction



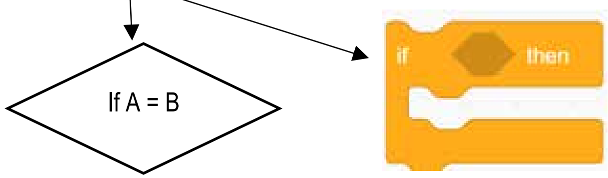
Activity 6 – open ended (they do their own thing with what they know – can also explore something new)
Plan, design and develop a block-based application of your choice, using the knowledge, skills and experience you have gained so far.
Amongst blocks, also need to use the Random block as well as the Ask and Answer block.

Note
Programming concepts are mostly abstract and intermediate phase learners still didn't reach the formal stage of cognitive development. Therefore, abstract thinking is still not reached.
It is therefore important to make concepts concrete and ensure that learners understand the concept well.

Introduce simple IF…THEN statement formally.
Learners need to understand:

- What a condition is
- How program flow is impacted by the outcome of the condition
- Compare to where they intuitively used an if, e.g. if on edge bounce

The diamond represents a decision based on a condition.

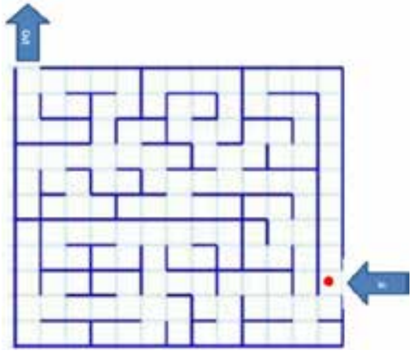


Compare two values.
Branch based on the outcome:-
yes or no / true of false
If outcome is no or false, instructions after the if-block are executed
If outcome is yes or true, instructions within the if-block are executed, then instructions after the if-block are executed.

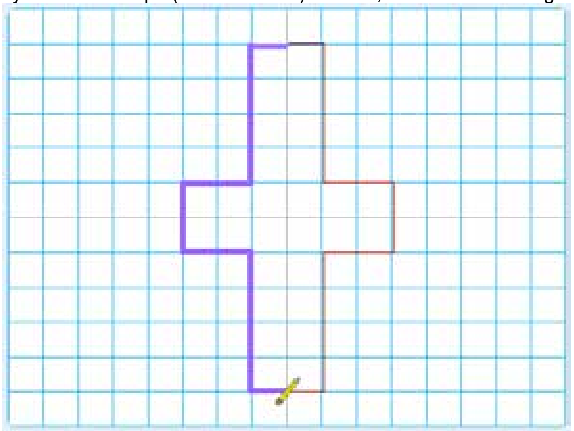
C.3 Interpret and execute a given symbolic or written set of commands

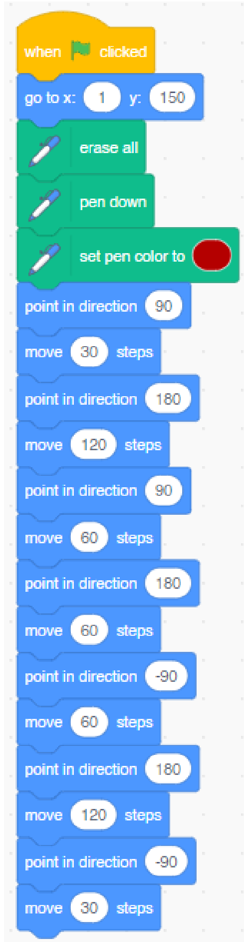
Link to C.1, C.2 and C.4 and link to D.10

Example activity 1 (pen-and-paper)
Vuyo needs to walk through the maze from the entrance to the exit.
Below are instructions someone started but did not complete:
Enter the maze (step onto the red dot)
Turn right, Walk 8 steps, Turn left, Walk 3 steps
Turn left, Walk 1 step, Turn left, Walk 2 steps
Turn left
Complete the instructions (algorithm) that will guide Vuyo through the maze successfully.



Example activity 2 – Link to D.10
Learners are provided with a stage, with a partial geometrical shape (purple line below).
The learners must then plan and write the algorithm and translate the algorithm to code to draw the corresponding symmetrical shape (red line below) to code, execute and debug until correct.



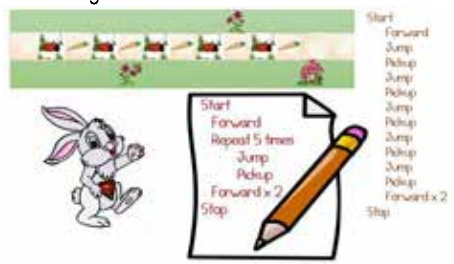


On worksheets (pen-and-paper-based activities)

- provide learners with algorithms and let them explain in plain language what the code does/what the output would be.
- Provide learners with block-based code, let them explain in plain language what the code does/what the output would be, then implement the code in the block-based app and discuss their initial interpretation of the code and where they might have misinterpreted the code and why.

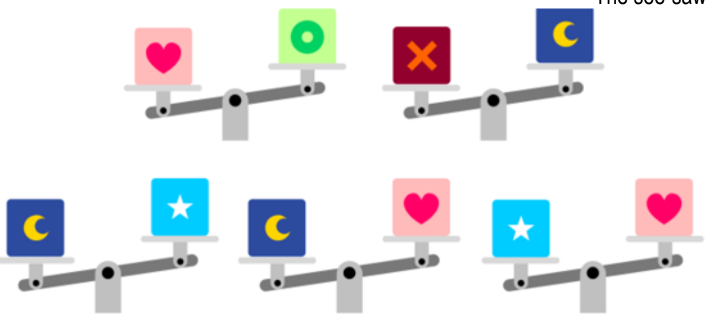
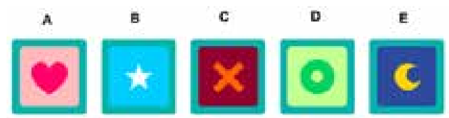
Note:
As a first step to scaffold learning, example activity 2 can be done physically on a grid or using pen-and-paper.

Note:
The role and use of suitable paper-based activities should not be neglected. In the example below pseudocode is written as simple instructions to solve a problem. SSB wants to collect all the carrots jumping over the logs.

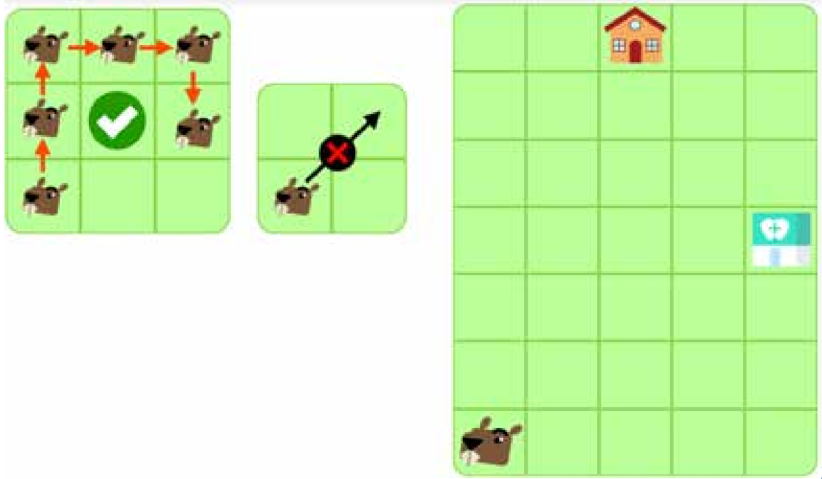


Note: The inclusion of e.g., a calculator in a mathematics curriculum does not exclude the requirement to still do and solve maths problems on paper. The same principle applies to problem solving and computational thinking in coding across grades.

C.4 Debug a given symbolic or written set of instructions	**Link to C.1, C.2 and C.3 as well as R.6 and D.10**

Example activity 1 – Debug Learners work individually. The teacher presents each learner with the set of code and explains that the robot is supposed to move 100 steps then turn 90' and then repeat this process four times till the robot has turned all the way around and moved in a "square". However, the code provided does not achieve the outcome. Learners need to debug the code: Trace the code using pen-and-paper, find the bug and correct it. Run the program to test if it is working. **Example activity 2** Provide learners with a problem and incorrect code to solve the problem. Learners then need to debug the code. 	Learners need to debug (find the error and fix the code to achieve the intended outcome) incorrect code. Learners can tinker and apply problem solving to "fix" the code
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to C.1**
Example activity 1 Five boxes have different shapes drawn on them. Compare the weight of the boxes with the aid of a seesaw.  The see-saw was used five times, with the following results:  Which box is the heaviest?  2021-TS-Elementary-Question-Paper.pdf **Example activity 2**	Learners use computational thinking to solve the problem Pattern recognition is part of computational thinking and is used to identify patterns in coding problems and/or data by identifying similarities or differences that can help to solve the problem or refine the algorithm.

Beaver is heading home from school, but first, he must make an appointment at the dentist. Beaver plays a game where he tries to complete the journey by only going straight ahead or turning right. He can do these two rules as many times as he wants and, in any combination, but he must NOT go diagonally.
Is it possible to reach first the dentist and then home if Beaver follows his rules? Figure out and choose the right answer below.



A. It is not possible to reach both places following these rules.
B. It is possible if he turns right exactly 4 times.
C. It is possible if he turs right exactly 2 times.
D. It is possible if he turs right exactly 6 times.

One of the main tasks of coding is to search for possible solutions, solutions that follow certain conditions. The question that is often asked is whether there is at least one possible solution.

2022-TS-Elementary-Question-Paper.pdf (olympiad.org.za)

Now write the algorithm for beaver heading home following his rules.

Robotics

R.1 Explain what a robot is in simple terms.

R.2 Identify different types of robots.

Briefly ask questions for learners to retrieve their knowledge about what a robot is. Then remind them of the two 'types' they learned about in term 1 (virtual and physical). Also remind them that they use a virtual robot when coding (sprites/objects).
Now, expand on physical robots:

Example activity

Provide each learner with a KWLS chart and ask them to write down what they know about robots and what they want to know (first two columns).
Learners watch a video what a robot is and expand on types of robots https://youtu.be/8wHJjLMnikU and complete the KWLS chart's columns on what they have learned about robots and what they still want to learn about robots.
Ask learners to report back on what they have written on the KWLS charts and facilitate a class discussion.
Show learners picture examples of different types of robots in different fields and briefly discuss what they do, e.g.

Field	Robot	Use
Medical	Paro (resemble a baby harp seal) 	**Therapeutic robot** (It shows lifelike movements and sounds) Designed to provide emotional support and companionship, particularly for individuals in need of social interaction or who might benefit from animal-assisted therapy. It exhibits lifelike movements and sounds. It has touch-sensitive sensors, that can make the whiskers move and that silently move its limbs and body. It also responds to petting by moving its tail and closing its eyes

R.1 and, R.2 can be done together

Learners need to acknowledge that robots are diverse and used for different purposes. Some use wheels to move, while others walk around on two, four, or even six legs. Underwater robots can swim, and drones can take to the skies.

There are robots the size of a coin and robots bigger than refrigerators. Some robots can make pancakes. Others can land on Mars.

Show learners examples for an overview of different types of robots – robots used for the following purposes: industrial (e.g. robot hand), service (e.g. vacuum cleaner), education (Lego-we-do), medical (e.g. therapy (Paro therapeutic tool) and exploration (e.g. drones).
Some background on the examples could provide context (no need that they must know all of this – just overview)

Robots play diverse roles in various sectors, leading to increased automation and improved efficiency.

Learners need to know that

- robots come in various forms shapes and sizes.
- robots are used in various industries.

Service (Home)	Vacuum cleaner  	**Cleaning robot** Using brushes to suck up dust and debris and transfer it to their built-in waste bin. They are object avoidance robots – sensors detect walls and obstacles and let them turn away. They also use sensors to detect dust and debris.	• specific robots perform specific tasks. and provide an example for each of the fields
Industry (Factory)	Robots assembling cars 	**Assembling robots** Could include cameras for sensors to 'see' Mechanics like arms that can move	
Education (Social robots)	Social robots in education  	**Learning assistants** Social robots can act as learning assistants to teach kids skills such as language, chess, etc.	

R.3 Outline the different components of a robot

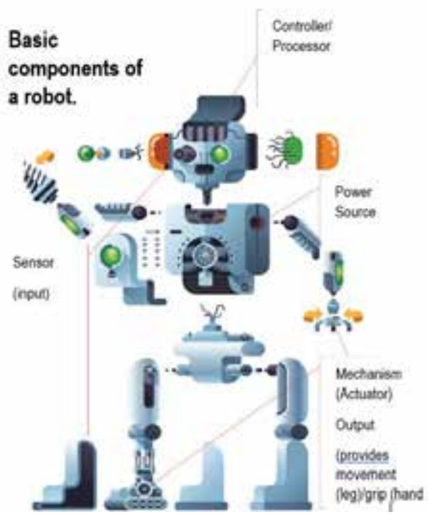
Example activity

Learners in pairs: The one learner is the 'robot' (follows instructions), the other learner is the 'controller'(processor) and gives the instructions. The 'robot' sticks out both hands with fingers up (sensor to feel). The robot can walk in any direction (walk towards a wall)

The 'controller' gives the instruction "start" to start walking and the 'robot' starts walking towards a wall. When the 'robot' touches the wall (with the hands (sensors), the 'controller' gives the instruction "stop" and the 'robot' stops.

Pretty much like we humans receive inputs from our sensory organs (like when the 'learner robot' touched the wall), our brains process the input (our brain tells us that we touched something hard), and we carry out the desired action (we stop) because our brain told us we could not go further); robots too have the same building blocks:

- Like we receive 'input' through our senses (see (eyes), hear (ears), feel (our hands), we smell (our nose), taste (our tongue), robots receive input via sensors.
- Like our brain processes the input we receive, for robots the processing is done by the controller (processor)
- Like we react based on the input received because the brain processes it and tells us how to react, and we react, e.g. by stopping, the robot's processor instructs the robot to react (output) in a certain way such as to stop.
- Like we need energy (food) to act, the robot needs power to perform.
- Like our bodies house our brain, arms and feet, the chassis houses the components of the robot.



Adapted from: Anatomy of a Robot | STANFORD magazine

So, we see that a robot has five main components: Sensors, processor (controller), actuators/mechanical mechanisms (for output), a power source and a chassis to keep the first four together.

Any robot is made up of three parts – Sensors (for input), CPU (processor), and Mechanical Actions (for output).

Link to R.1 and R.2

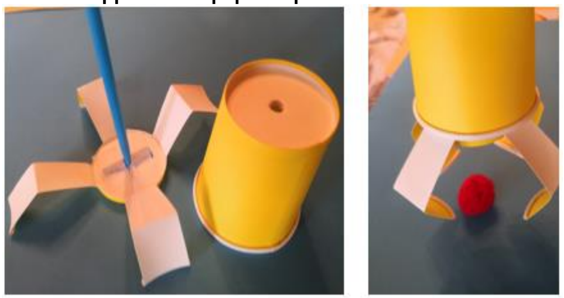
Learners need to list the following man components of a robot and briefly outline the function of each.

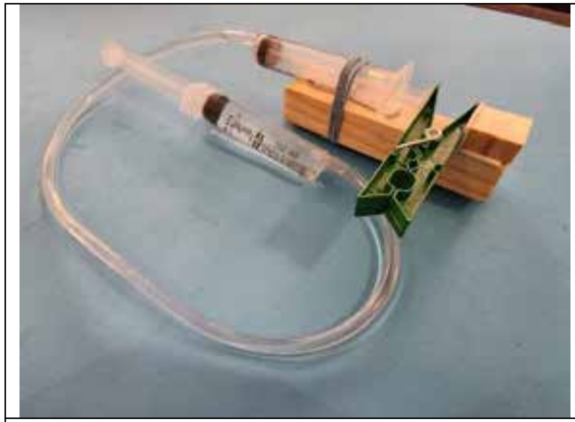
Robots are amazing machines that use sensors, controllers, and actuators (mechanisms/mechanical actions) to do their jobs. Sensors are like their eyes and ears, helping them see and hear what is happening around them. They can sense things like temperature, pressure, and can move, grab things, and do their jobs well. movement, which gives them vital information. Controllers are like their smart brains. They use the information from the sensors to make decisions and give commands. And actuators are like their strong arms and legs. They follow the commands from the controllers to do tasks and interact with the world. All these parts work together so robots.A Simple Explanation of How Do Robots Work - Tech Spirited

At an elementary level, learners need to know that a robot is made up of the following main components:

- Sensors (for input)
- Controller (for processing (processor like a computer))
- Mechanical actions (for output) (actuators))
- Power source (e.g., battery)
- Chassis that houses all the above

and the basic role of each

Now, each pair draws a picture of a robot, showing the parts. Refer to the types of robots and the types they learned about in Term 1 and discuss.	
R.4 Present an understanding of how robots affect the world	**Link to R.1, R.2, and R.3**
Learners have now learned the basics of what a robot is, examples of different types of robots, what the basic components of a robot is and some applications (how the affect the world) of robots. Briefly revise the above concepts and extend on how they affect the world from Term 1 and R.2. Use the examples used in R,2 term 2 (types of robots), divide learners in small groups (not more than four), give each group, on a worksheet, one of the example robots (picture) and let learners discuss and write down what they think the impact of the specific example is (possible positive as well as negative impact). Random groups report back, and teacher write the main points on the board. Teacher concludes by summarising the impact of each.	Robots have a significant impact on the world, touching various aspects of society, economy, and daily life. **Industry:** Robots have revolutionized industrial processes by taking over repetitive tasks such as assembling cars.
R.5 Design a simple artefact based on a set of design instructions	**Link to R.1 – R.5 and R.6**
Example activity – End effector - Gripper Concept of end effector: • An end effector is an important part of a robot that helps it interact with the world around it. Think of it as the "hand" or "tool" of the robot. Just like we use our hands to pick up objects or perform tasks, the end effector is what the robot uses to do its work. • The end effector can come in different shapes and sizes depending on the robot's purpose. It can have things like grippers, claws, or specialized tools attached to it. For example, a robot in a factory might have a clamp-like end effector to pick up and move objects. Another robot in a laboratory might have a small arm with a precise tool for conducting delicate experiments. • The end effector is usually located at the end of the robot's arm or manipulator. It can be controlled by the robot's computer or by a human operator. The robot can use its end effector to perform tasks like picking up objects, assembling parts, painting, or even playing games. Learners first need to design the project, using design thinking and the engineering design process. **Build 1 Gripper with a paper cup**  **Build 2 – Improvement using masking tape, straws, string and two paper cups.**  **Example activity 2 – Hydraulic gripper**	Learners design and make a robot gripper that will pick up paper and put it in a recycle bin. The gripper project must enable learners to acknowledge, at an elementary level, the mechanism of the gripper and how it is controlled - a key concept in robotics. Explain the concept of Explain the concept of an end effector. In simpler terms, the end effector is like the robot's hand that allows it to interact with its environment and perform different tasks. It's an essential part of the robot that helps it be useful and perform its designated job. The following gripper could also be considered: (gripper made with ice cream sticks and filing-split pins. Holes punched using a paper hole-punch).



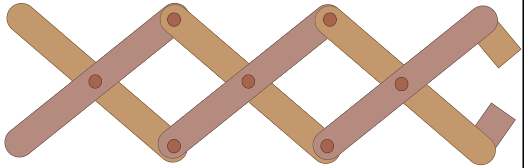
Example 2 illustrates a gripper using a basic hydraulic system with an action and reaction. The water is displaced from one syringe to another causing the peg to open.

Expansion on the Grabber – With unplugged activities.

The following is an example of a design-based Coding and Robotics, unplugged activity. The learners are expected to build a grabber based on a design and then improve on it.

Thereafter the grabber is used as an educational tool in a coding activity that simulates a realistic robotic operation.

The basic design of the grabber mechanism is schematically represented on the right:

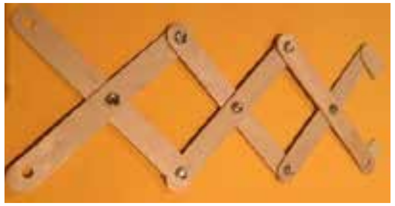


Six wide bamboo (or hard carton) ice cream sticks are used with 7 split pins and two small pieces of off-cut ice cream sticks.

The hole markings on each stick are measured to ensure that there is a hole in the middle and equally spaced holes at the end of the stick. The two off cut grabber pieces can also be marked.

A punch used upside down without the cover, which allows for the easy alignment and punching of holes.

(Note not all sticks need to be cut with three holes)



This improved grabber (see ***Notes*** in right column) can then be used to mimic the "sorting operations of a robot in a factory".

In the example below a 1 x 4 grid represents a "factory floor". Pom poms in two different colours are placed on the first part of the factory floor. A robot with a grabber arm will move to the first block and then needs to sort the pompoms into the correct paper cup bins.

Note:

The effectiveness of the grabber can then be tested to see if simple objects such as pom-pom's, cotton wool balls, paper cups etc can be lifted and "moved".

The learners may want to redesign the grabbing portion by including other grabbing pieces such as paper clips, smaller additional sticks, or carton attachments etc.

As part of the activity the learners may also be posed the question: How would you improve the design of your grabber? (This open-ended question allows the learners to think and any appropriate answer such as:

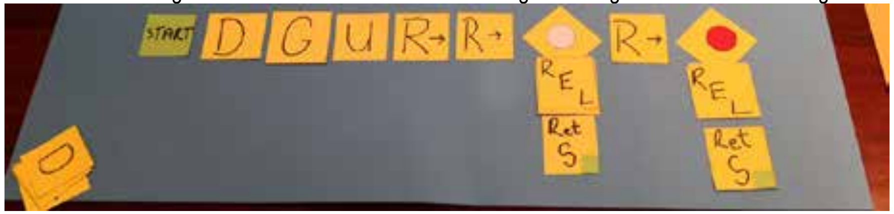
"The grabber is made of wood, maybe plastic will be better?".

"The split pins could be replaced with a simple bolt and nut."

In addition, the learners could also be shown an alternative design of a grabber and do an evaluation of the two.

In the alternative design a rubber band is included and the first part of the grabber, the middle joint is connected with a split-pin to a sturdier and bigger "ice cream stick". Pulling on the string created tension and releasing the string releases the grip as well.

The robot arm will move to the start position above the pom poms.
Learners use coding cards with different commands to design a coding solution to control the grabber to sort the pom-poms into the correct cup.



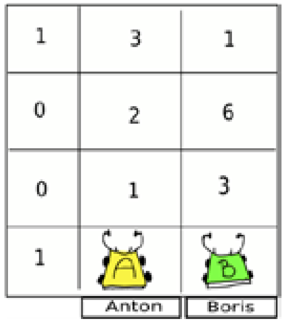
The following actions are represented in each of the cards where the

- START indicates the start of the code/operation
- D states that the grabber should go down.
- G indicates grab
- U indicates the grabber should go up one position
- R→ indicates that the grabber should move right one position.
- The diamond card represents a condition (if the pom-pom is white) then
- REL – Release
- Ret S which states return to start position.

The coding cards represent a solution to pick up one pom-pom and place it in the correct cup.
Thus, for three pom poms the learners could recognise (if this activity is done at a later grade e.g. grade 6 that a repetition structure could be included) To repeat the instructions whist there are still more pompoms left.

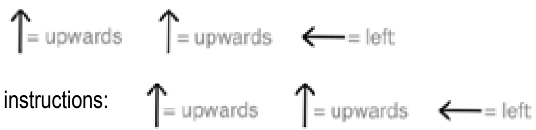
R.6 Mimic the operations of a robot	Link to R.5 and C.1, C.2, C.3 and C.4

Example activity 1 (link to R.5 – gripper)
Learners use the gripper to pick up objects and observe how it works in picking up something.
Example activity 2
The rectangle on the right shows the map of a park divided into sections. The number in each square tells you how many pieces of trash visitors left in that section of the park. The park rangers have two robots, Anton and Boris, that collect the trash they find in every section they enter.



Anton was sent first with the following instructions: ↑ = upwards ↑ = upwards ← = left

Once Anton was done, Boris was sent with the same instructions: ↑ = upwards ↑ = upwards ← = left



How many pieces of trash will Boris collect?

[2019-TS-Elementary-Question-Paper.pdf (olympiad.org.za)]

At an elementary level, learners must be able to acknowledge how the gripper works and is controlled to pick up an object.

Object/Sprite in block-based coding application is a virtual robot
Writing instructions for the virtual robot/sprite to move on the grid

Digital Concepts	

<table>
<tr><th>D.1 Outline the concept of technology and purpose of information technology (IT)</th><th>Link to D.4, D.5 and D.6</th></tr>
<tr><td>Revise the concept of technology by asking the learners if they can remember what "technology" means and write down their answers on a board. Remind them that technology refers to any tool or invention created by humans to solve problems and make tasks easier. Revise the concept of information technology, reminding them that it specifically deals with the use of computers and software to manage and process data and information and solve problems.
<u>Example activity</u>: Distinguishing between Technology and Information Technology
Provide a handout with pictures about technology, including its definitions (such as the wheel, computers, smartphones, cars, etc.).
Divide the learners into three groups and provide each group with a worksheet with two empty circles that intersects. The one circle is named Technology, and the other circle is named Information Technology.
Ask the groups to identify which pictures belong to technology, information technology and which intersect.
Display the learner's visual representations and discuss their similarities and differences.</td><td>Reinforce and extend from Term 1 using different activity.
Technology refers to any tool or invention created by humans to solve problems and make tasks easier.
Information technology specifically deals with the use of computing devices (hardware) and software to manage and process data and information and solve problems.
Discuss similarities and differences.</td></tr>
<tr><th>D.2 Recognise that he or she is living as citizens in a digital world.</th><th>Link to D.6</th></tr>
<tr><td><u>Example activity:</u> Design a good digital citizenship "road" sign that digital citizens will obey.
Link digital citizenship to good citizens that obey road signs. Show examples of road signs and ask learners what road signs they see on their way to school and why one needs to obey them. The teacher leads the discussion around signs (instructions) that humans and robots can use.
The teacher makes learners aware of what good digital citizens do and makes a list of rules, e.g.
Do not share a password/pin (use your head to think about your safety)
Do not be a bully (use your heart and care for others)
Do not share private information (use your head to think about your safety)
Do not talk to strangers (use your head to think about your safety)
Balance time spent on digital devices (use your arms to balance yourself)
Care for your devices (use your heart to care for your devices)
Stay safe online (use your head and think about your safety)
Beware of what tracks you leave online (use your feet responsibly)
Be kind and respectful online (use your heart and care for others)
Don't start a fight online (use your heart, head and your feet)
While the teacher discusses the rules, each learner writes a rule on a strip of paper and throws it into a box.
Learners are then divided into pairs, a navigator (gives instructions to the driver) and a driver (carries out all instructions given by the navigator and draws the sign).
Each pair draws a rule from the box and create a good digital citizenship "road" sign.
Pairs take turns, hold up their signs and pledge to obey the rule that appears on their sign.</td><td>Reinforce from Term 1 the concepts of concepts of digital world and digital citizenship – what it means.
The online world also comes with its own dangers and challenges such as security and privacy issues. We therefore need to use passwords/pins to protect our access to devices/private information.
Refer to online bullying.
In the online/digital world, we all leave a digital footprint (just as the one we leave when walking on sand)
The digital footprints may let others know things about us that we do not want them to know which could misused by people to bully us.

Provide a brief overview (to create a basic awareness) of the above and briefly discuss (overview) the aspects of good digital citizenship (detail are dealt with in later terms and grades).

Do with D.8 – each digital citizenship "road" sign communicates a message.</td></tr>
<tr><th>D.3 Demonstrate an understanding of the concept of a computing device.</th><th>Link to D.1, D.5, D.7</th></tr>
<tr><td>Briefly explain the basic concepts of hardware and software by asking learners what they remember by the concepts of “hardware” and “software”. Refer to D.1. Ensure learners understand the difference between the two and their features.
Example activity:
Classify the following according to hardware or software by making a cross in the hardware or software row:
<table>
<tr><th></th><th colspan="6">Technology (Hardware and Software)</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Hardware</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Software</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td><td>Reinforce and extend from Term 1 using different activities.
The components of a computer system are made up of:
Hardware which are the physical parts of the computer, for example, input devices, output devices, Central Processing Unit (CPU) and storage devices. Hardware is also the electric, electronic and mechanical parts of a computer.
Software is the computer programs or applications, used to perform a specific function.

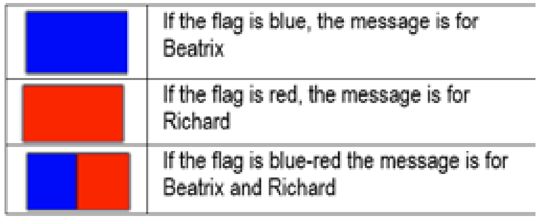
Also link the concept of software to the coding app that they use and the concept of hardware to the device they need to use the coding app.</td></tr>
<tr><th>D.4 Identify the common uses of ICT in the real world</th><th>Link to D.1 and D.3 and D.5</th></tr>
</table>

Linking to what ICTs are and what the different components of an ICT system is, briefly explain to learners that IT mostly deals with computing and data and information management, whilst ICT adds 'communication' – being able to exchange data and information over networks. ICTs are therefore used to communicate with people who all over the world. Daily, we can use ICT to send WhatsApp messages, emails, and make phone calls. It is also used to store information like pictures, videos, and documents. We can use ICT to find information on the internet. We can also use it to learn new things like math, science, and history. In schools, teachers use ICT to teach learners by showing them videos or pictures on a computer or whiteboard or digital projector. Ask learners to provide more examples of ICTs in their daily lives and discuss these.	Done in relation to D.5 (first do D.5 then do D.4) ICT is broader than IT. IT includes 'communication'. As computer networks became more prevalent and the internet revolutionised communication and data sharing, the scope of IT expanded to include these communication aspects, leading to the term ICT. Learners need to know: • What ICT is • How it differs from IT • Identify some common uses in their daily lives
D.5 Differentiate between the components of an ICT system	**Link to D.3, D.7**
Computers and Devices: They are the computers, laptops, tablets, and smartphones we use (like team members in the system) **Internet**: Allows all the computers and computing devices to communicate and send information and data (Like a magical road that connects all the team members (computers and devices together)) **People**: We are the most important part of the ICT system! We use the computers and the internet to do so many exciting things. We can talk to our friends and family, explore fun websites, and learn new things. The sales point in the shop has a scanner that reads the barcode on the item and adda the price of each item to give you the total amount payable. Another part, the card machine reads your banking details and make a payment (use examples that learners understand) Parts include: • Hardware (input and output devices), e.g., till, barcode reader and the card reader • Software (code) – programs that enable the system to work. • Data that is processed and stored, e.g., read barcode on items to get prices and calculate amount due • The Internet (network) that communicates with the bank to make a payment / communication between till and barcode reader or the card reader. • People that operate the devices and users that communicate with others using ICT systems. ICT system is like a big team of computers, devices, the internet, and people working together to make our lives better and more fun! **Example activity:** Differentiate and sort the components of an ICT system. Set up a designated wall space for sorting the components. Divide the wall into sections for each component of the ICT system. Divide the class into groups. Each group will receive a set of cards with pictures representing the various components of an ICT system. Ask each group to sort their cards and place them on the designated sections of the poster board or wall. Explain the correct placement of each component, providing further clarification or examples where needed for learners to have a clear understanding of the different components. Elaborate on the interactions and relationships between the components within an ICT system.	Learners need to understand, at a basic level, that: An ICT system is made up of computing devices (e.g. computers), programs (the instructions that tells the devices what to do), data and information and networks (including the internet) that allows the devices to communicate and send data and information as well as the people that use all of this. Components of an ICT system • Hardware (e.g., computers) • Software (e.g., operating systems, applications, programs) • Data (e.g., information) • Networks (e.g., internet) • People (e.g., users)
D.7 Present a basic understanding of the concept of input processing and output.	**Link to C.2- C.5 and D.10**
Examples: **Simple, everyday example: Cooking an egg.** Egg (input) → boiling in water (processing)→ cooked egg (output)  **Using the point of sales ICT system** (shopping) The barcode scanner linked to the cash register (hardware) inputs the item codes, then the processor of the cash register processes the item prices and then provides output in the form of the amount payable (on screen) and a slip (on paper) using a printer (output device). The person that operates the pay point (till – hardware) is also part of the ICT system; so is the code (software instructions) that calculates the prices. Also use other examples such as • A program they write in the block-based application – when they click the green flag (input), the code is processed (executed) and there is some result (output) such as the sprite/object doing something. • What they have learned from their understanding of the components of a robot e.g. (sensors (input), controller (processing), movement (output))	Present learners with various examples of input-process-output, starting with an everyday example, then moving to ICT examples and programming examples Learners need to acknowledge that, when working with IT or ICT systems, there are input (from input devices such as keyboard, mouse, scanner), processing (processor) and output (from an output device such as a screen, printer, speaker)

<table>
<tr><th>D.8 Interpret a pattern to represent or communicate a message or image</th><th>Link to C.1, C.2, C.3</th></tr>
<tr><td>

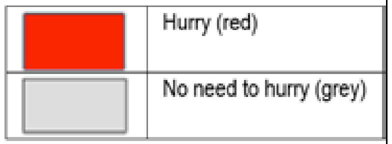
Example 1 Interpret messages
Albert is the father of Beatrix and Richard. Albert is a guard on the local beach.
When Beatrix and Richard are playing on the beach, Albert uses flags to send them messages.
Below is a description of what the flags mean:

Top Flag



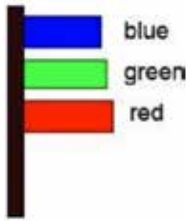
Middle Flag

Bottom Flag



Answer the following questions:

- What do the flags on the pole on the right mean?
- Draw a pole with three flags (as indicated above) that will send the following message:
  - Beatrix and Richard, drinks available, no need to hurry.
- Now, in pairs, each learner compiles their own message and draw a pole with three flags (as prescribed above), then show the message to their friend to 'read'/interpret



2017-TS-ELEMENTARY-Q-paper.pdf (olympiad.org.za)

Example 2
Learners create road signs communicating 'rules' for being a good digital citizen (link to D.2)

</td><td>

Note:
Example 2 can be done with D.2

</td></tr>
<tr><th>D.10 Demonstrate a basic proficiency in the application of digital skills.</th><th>Link to C.2, C.3, C.4 and C.5 and D.3 and D.7</th></tr>
<tr><td>

Elementary file management – create their own folder to save their coding programs, using an appropriate name for their folder.
Open files from and save files to their own folder.
File names: saving the file using a meaningful file name so that it is easier to identify and retrieve at a later stage.
Highlight the concept of a file extension – indicates to the computer which program to use to open the file.
When they need to save the coding programs they created (C.2 and C.3), explain the process of input, processing, output and storage and the computer parts/ devices involved and link to D.3 and D.7 (initially, saving their work is initially incidental learning as learners will use the default folder to save their programs.
Also, when they need to open their saved program files, explain the concepts of input, processing, output and storage again.

</td><td>

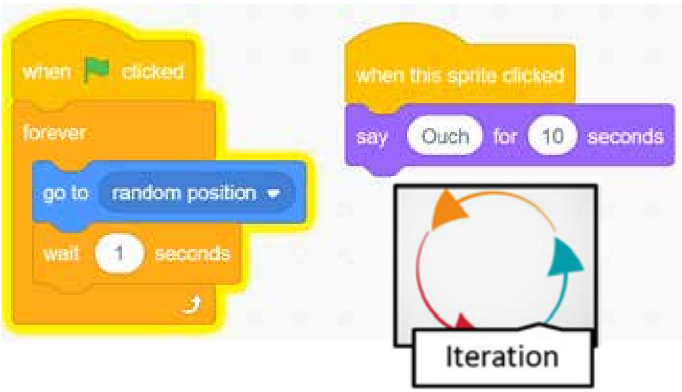
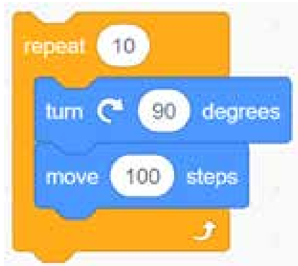
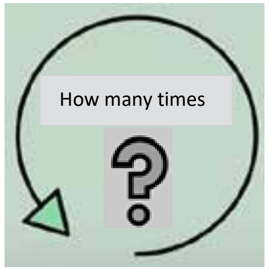
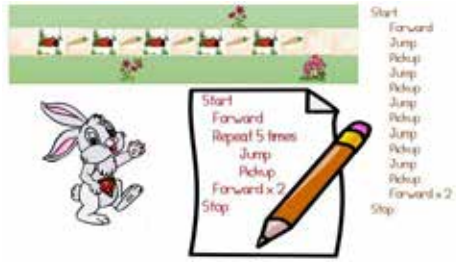
Revise and extend from term 1.
Learners must be able to do elementary file management on the computing device:

- Create a folder with their name for saving their block-based programs.
- Retrieve and open their saved programs from their folder.
- File naming conventions
- Importance of file extension (only .sb3) (or .sb2/1 if earlier versions ore used)

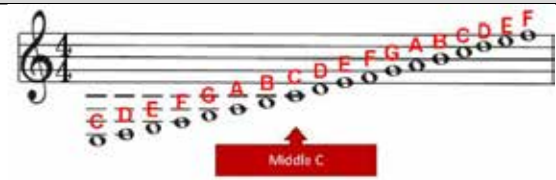
Learners must also link file management to input-processing-output concepts

</td></tr>
</table>

3.1.3 Term 3

Content (Grade 4 / Term 3)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.2 – C.4 and C.6 and R.1-R.6**
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C.1 and D.10** **Done with C.1**
Example activity 1 Write a program that does the following: A sprite moves to a random position forever If you click on the sprite, it says 'ouch' Solution on the right.  **Example activity 2 – Introduce Repeat with fixed number** Use Example activity from Term 1 (draw square) Learners must identify the pattern and use a repeat construct to rewrite the code for drawing a square New code using a Repeat 4 times to replace sequential code 1. Turn right. 2. Draw a 3 cm line. 3. Turn right. 4. Draw a 3 cm line. 5. Turn right. 6. Draw a 3 cm line. 7. Turn right. 8. Draw a 3 cm line.  **Note:**  Loops also ask questions, such as *how many times?,* but ask the question over-and-over again and perform actions over-and-over until the condition is satisfied **Example activity 4 Open-ended** Using computational thinking, learners design a game with the knowledge, skills and experience gained up to now. Concepts they could include: random position, repeat, if on edge, bounce, when sprite is clicked, using a grid to find a treasure, etc.	Introduce Repeat with fixed number (constant value) Learners generally also struggle with loops, therefore, in Grade 4, only the repeat with a fixed value is introduced to scaffold the concept of loops. Learners first need to identify the pattern to be repeated (computational thinking) **Note:** Generally, most learners are comfortable with tasks/problems that require them to write code that requires them to combine one or two concepts at a time. Many learners, however, struggle when they must combine many coding concepts at a time/in one program as it increases the difficulty level as well as the complexity of the task/problem. It is therefore advisable that leaners practise coding concepts using small, basic, manageable tasks/problems until they are ready for the next step. Practical paper-based activities can be applied to strengthen the mastering of the content. In the example below pseudocode is written as simple instructions to solve a problem. SSB wants to collect all the carrots jumping over the logs and equivalent solution is presented using code blocks. 

Content (Grade 4 / Term 3)	Notes/Examples
	
C.3 Interpret and execute a given symbolic or written set of commands	**Link to C.1, C.2 and C.4**
Example activity 1 – Answer block Provide learners with the code on the right using a worksheet (paper-based) Learners need to study and interpret the code and explain what it does and predict the output. After explaining, learners now compile the code in the block-based environment. Learners then run the code and compare it to their interpretation and predicted output   **Example activity 2 – IF…THEN with Answer** On a worksheet, provide learners with the code on the right. Learners need to study the code and explain what it does. Then answer the following questions about the code: • What will happen if the user enters 4 (the user is in grade 4)? What output would the program give? • What will happen if the user enters 5 (the user is in grade 5)? What output would the program give? Learner can now implement the code in the block-based environment, enter 4 and 5 as input and compare their answers with what they expected it to be. If their answers differed from the output when the code is implemented, they first need to check if their code is correct. If correct, they need to ensure that they understand why the output differ and where their reasoning might have been wrong.  **Example activity 3 – Open-ended** Using computational thinking, learners design a game with the knowledge, skills and experience gained up to now. Concepts they could include: repeat, random position, if on edge, bounce, when sprite is clicked, ask & answer using a grid to find a treasure, etc.	**Introduce Answer-block to learners** The answer block is a sensing block and a reporter block. It reports the most recent text/value inputted with the ***Ask and wait block***. The main purpose of the answer block is to store the answer (that was typed by the user in the input field) and, if needed, it can also display it on the screen. **Note:** The answer-block should not be seen as a variable (though it 'holds' the most recent text/value inputted) as it reports specific things/most recent inputted while variables can be changed to whatever you want/ through code) ***Variables are only introduced in Grade* 6** as, generally, learners struggle with conceptual understanding of introductory programming concepts such as variables, expressions, and loops (Grover *et al*, 2019). Learners, therefore, only need to understand the concept of keeping a value, e.g. 'answer' getting 'something' from the user and 'keeping' it to use or display it. **Note:** It is important to engage in pen-and-paper activities where learners need to study code, explain what it does / provide the output of the code

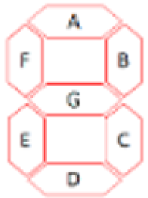
Content (Grade 4 / Term 3)	Notes/Examples
Example 4 – Interpret code and fill in the missing code The girl sprite wants to greet the user by first asking the user's name, then greet the user using the name as illustrated below using the code on the right   Study the code and fill in the missing code that will result in the above output	
C.4 Debug a given symbolic or written set of instructions	
Example activity 1 Provide them with incorrect directions from the class to the principal's office. The need to find the error and correct it and test it to ensure that any person (even someone that does not know the environment/where the principal's office is) that follows the corrected directions will be able to reach the principal's office.	Provide learners with incorrect algorithms (pen-and-paper) as well as incorrect block-based code (programs) and a description of what it should do/what the outcome should be, which they then need to correct.
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	
Example activity Music follows a pattern. Use the information/pattern on the right to complete the names of each music note by writing down the names of each music note as indicated for the Bana ba Sekole song below (the first six is already done)    Whole Note / 4 Counts / Whole Rest / 4 Counts Half Note / 2 Counts / Half Rest / 2 Counts Quarter Note / 1 Count / Quarter Rest / 1 Count Eighth Note / ½ Count / Eighth Rest / ½ Count 1/16th Note / ¼ Count / 1/16th Rest / ¼ Count 1 Count = 1 Beat	Also provide coding problems where learners need to identify patterns / code repetition **Note:** It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively. By identifying patterns, we can predict what will come next and what will happen again and again in the same way. In Computer Science/coding we analyse patterns in data and make predictions and generalisations based on the pattern analysis.

Content (Grade 4 / Term 3)	Notes/Examples

C.7 Create or complete a pattern to represent a data set

Example activity

A calculator displays numbers using the following pattern:



For example, the number 2 is displayed switching on the following segments must be on:

A B G E D

1. Complete the table for all the other numbers following the pattern above.
2. Write down the segments for displaying a capital letter C and a lowercase c
3. Using the 0s and 1s that represent the letters, write the word ***Hallo***.
4. See how many letters from the alphabet one would be able to display.

Digit	Segment A	B	C	D	E	F	G
0							
1							
2	1	1	0	1	1	0	1
3							
4							
5							
6							
7							
8							
9							

Notes/Examples: Also provide coding problems where learners need to use patterns / loops for repeated code

Robotics

R.1 Explain what a robot is in simple terms.

Notes/Examples: **Link to R.2, R.3**

R.2 Identify different types of robots.

R.3 Outline the different components of a robot

Briefly revise what a robot is and let learners identify different types or robots. (from previous grade and terms)

Ask learners to write down what they think the different components of a robot is.

Hand them a worksheet with questions that they need to answer while/after watching the video.

Now, let them watch the video https://youtu.be/CrQ5atmjSqQ

After completing the worksheets, ask random learners to provide their answers to the questions and discuss their answers.

Consolidate using the following example:

If a robot is built to move freely in any direction but to stop once it bumps into any object):

- Receive power (from e.g. a battery) to start working/moving
- Input: via a sensor that detects when the robot bumps into an object
- The processor (controller) will process the bump action and 'instruct' the robot to perform an action (output)
- Output: stop the motor (mechanical action)
- Chassis

Once the robot bumps into an object, its input sensor (touch) will be activated (turned on). This sensor will send signal to the processor when it turns on. The processor will look up in its list of instructions to find the relevant action to be performed upon the reception of this signal.

Notes/Examples:

Note:

R.1, R.2 and R.3 can be done together

A Simple Explanation of How Do Robots Work - Tech Spirited

Revise and extend from previous terms.

Extend to different types of sensors:

Learners need to know (at an elementary level):

- The sensory inputs that the robot takes can be anything from smell, touch, visual differences, etc.
- The central processing unit is the microprocessor or microcontroller that processes this input, searches for the corresponding function to perform from an instruction set, and then sends the signal on to the output mechanism.
- Upon reception of this signal, the robot will perform the desired action.

R.4 Present an understanding of how robots affect the world

Notes/Examples: **Link to R.1 – R.3**

<table>
<tr><th>Content (Grade 4 / Term 3)</th><th>Notes/Examples</th></tr>
<tr><td>

Example activity – Class discussion of how robots affect the world we live in.
Aspects to discuss:
Robots are amazing machined that can help and make our lives better.
Compare the role of robots and people in the real world doing the same task. Consider aspects such as saving time, making work easier, helping people, exploring new things, entertainment)
Robots make our lives easier: Robots help grown-ups do chores and tasks faster, so they have more time to spend with their families and do things they enjoy.
Healthcare Heroes: Robots can help doctors and nurses take care of sick people. They can even do surgeries and help find cures for diseases
Discovering New Places: Robots explore places humans can't go, like deep oceans or faraway planets. They send back pictures and information to help scientists learn more about our world.
Building Things Better: Robots help make things in factories, like cars and toys. This means we can have more things that are made just right.
Learning and Fun: Robots can be our friends and help us learn new things. They can play games with us and teach us cool stuff.

Delivering groceries→

</td><td>

Note:
Can be done with R.1 and R.2 if time allows

Robots are machines that can do different tasks on their own without any help from humans. They can be very helpful and have an impact on our world in many ways.

</td></tr>
<tr><th>R.5 Design a simple artefact based on a set of design instructions</th><th>Link to R.4 and R.6 and R.7</th></tr>
<tr><td>

Example activity – Movable artefact with DC motor - Wigglebot

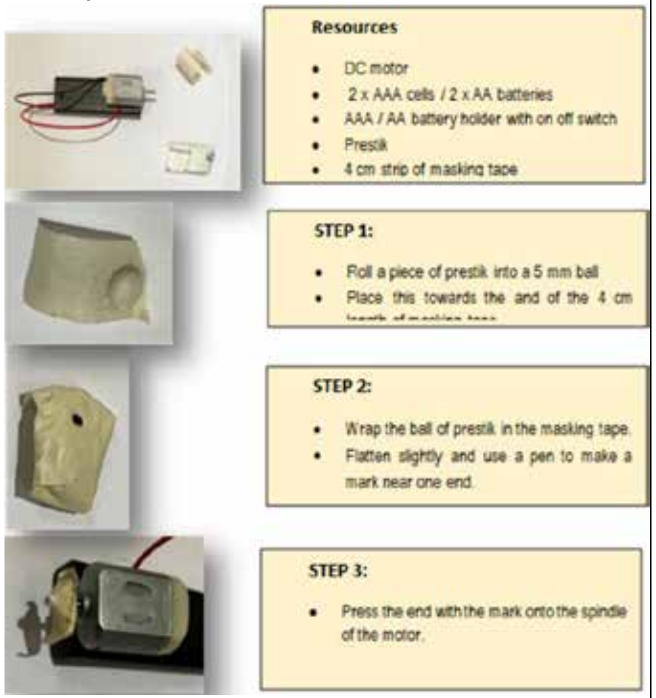
Final product

Learners use design thinking and follow design process to build the robotics artefact.
With a wigglebot the off-centre placement of a weight on a DC motor shakes the structure to make it move around.
Explain that motors in robots make them move:
Motors use electricity to spin and create motion. Robots have motors placed in different parts, like wheels or arms.

- If a robot has wheels, the motor makes them turn.
- If it has an arm, the motor controls its movement.

The motors are controlled by signals that tell them how to move, like going forward or lifting an arm.
Note:
There is no soldering involved in this project.

Steps

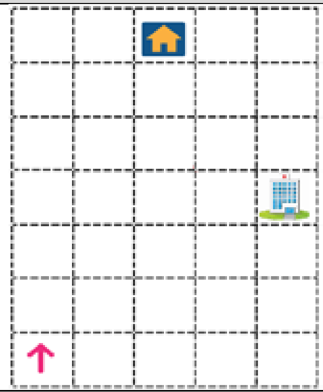


</td><td>

At and elementary level, explain to learners the robotics concepts used.
Note:
In some instances, depending on the context, one can provide learners with pre-packaged kits to practise their building skills, e.g. Project built from a KIT (Adapted)

Using a pre-packaged kit also has educational value as it typically includes sets of instructions for the assembly. This requires the leaners to apply the steps (i.e., follow an algorithm). In many cases debugging or corrections are also required. The learners can then be tasked to add alterations and or improvements to the artefact

</td></tr>
</table>

Content (Grade 4 / Term 3)	Notes/Examples
R.6 Mimic the operations of a robot	**Link to R.5 and do in relation to Cs**
Example activity 1 • On a grid, act out the algorithm created in Term 2 C.6 in activity 2 (debug if necessary) • Someone translated the algorithm into a block-based coding language but made a mistake (remember one can only go forward and turn right). Correct the error so that the sprite (arrow) can first visit the dentist and go home following the rules (sprite must end on the block with the home (middle block, top row)  **Activity 2 – open ended** Learners use block-based programming environment to create a robot sprite with instructions that mimics the operations of a robot.	**Note**:(Activity 2) Object/Sprite in block-based coding application is a virtual robot. Writing instructions for the virtual robot/sprite to move on the grid
R.7 Create, test and execute a set of robotic instructions	**Link to R.5 and R.6 and C.1 – C.7**
Use computational thinking, design thinking and the engineering design process to plan, build, test and debug a robotic artefact – the speedster robot (R.5)	Done with R.5 and R.6 When learners design and build robotics artefacts, they use computational thinking and design thinking and follow the engineering design thinking process which include executing, testing and debugging.
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.2 – D.7 and D.10**

Revise the concept of technology by asking the learners if they can remember what "technology" means and write down their answers on the board.
 	
Remind them that technology refers to any tool or invention created by humans to solve problems and make tasks easier.
• Facilitate a class discussion on:
    o how technology have impacted their lives
    o how their lives would have been without technology
    o what potential impacts technology could have on society?
their ideas and predictions for technology in the future.
	Do with D.6

D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.6**
Start a discussion about the dangers they experience in their world and how they should behave to avoid the dangers. Continue the discussion to make a list of the potential dangers they may encounter in the online world and create safety tips to make safe and responsible choices while using the internet. **Example activity:** Safety tips for online dangers Divide learners into different groups to create a poster about safety tips for online dangers. Each group is assigned one online danger and will be asked to create safety tips to avoid becoming a victim of the online danger. Ask learners to create a poster with the safety tips they discussed about the online danger assigned to them. Group present their safety tips to the rest of the class.	The online world also comes with its own dangers and challenges such as security and privacy issues. We need to use passwords/pins to protect your information. Dangers on the internet: (provide a brief overview) • Cyberbullying: Mean messages or threats online. • Stranger Danger: Avoiding communication with unknown people online. • Personal Information: Not sharing private information with strangers. • Inappropriate Content: Understanding what content is safe to view and what to avoid. • Online Games and Friends: Being cautious about sharing personal information or meeting online friends in person.

Content (Grade 4 / Term 3)	Notes/Examples
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.1 – D.5 from previous terms and R.1 – R.5**

<u>Example activity:</u> Differentiate between Technology, Information Technology, ICT

Divide learners in small groups (not more than 4).

Remind them that technology is all around us (and has been for centuries), and it includes all the tools and machines we use to make our lives easier and better. It can be something simple, like a pencil or a bicycle, or something more complex, like a computer or a smartphone. Technology helps us do things faster, communicate with others, and learn new things.

Use the evolution of the following technology: (technology à information technology à information and communication technology) as basis for discussion in groups and provide each group with the following pictures:

Technology	Information Technology	Information and Communication Technology (Allows communication through networks)
		

Teacher gives background on

- typewriter and how we communicated by typing letters and sending it via post.
- First computers (without internet)
- Today, with computing devices that communicate via internet.

Learners need to make a table with 3 columns – one for T, one for IT and one for ICT. In each column they write what people could/can do and could not/cannot do with each.

Notes/Examples:

Focus on the evolution from technology (T) to information technology (IT) to information and communication technology (ICT)

Remind learners:

Technology can be anything that makes our lives easier (e.g., electricity)

Information Technology (IT) is a special kind of technology that focuses on computers and how we use them to process and manage information. IT includes things like computers, laptops, tablets, and the software we use to create documents, play games, and do many other things. IT helps us store and organize information, like pictures, videos, and documents, so we can access them whenever we need them.

Information and Communications Technology (ICT): Now, let's add one more word to our technology journey: Communications. Information and Communications Technology, or ICT, is a big idea that combines Information Technology with how we communicate with others. It's like bringing together computers and other devices, like smartphones and the internet, to help us talk to our friends and family, even if they are far away.

ICT lets us do exciting things, like sending messages instantly to people anywhere in the world, making video calls to see and talk to our loved ones, and sharing our ideas and creations with others online. It's like having a magic box that connects us to people and information all around the globe!

In summary, they need to understand that it started with basic technology, which includes various tools we use (mostly non-digital). Then we moved to IT, which focuses on computers managing information. Finally, we reached ICT, which combines computers and computing devices to help us communicate with others and access information from faraway places. Technology keeps getting more amazing, and ICT is one of the coolest parts of it.

Content	Notes/Examples
D.7 Present a basic understanding of the concept of input processing and output.	**Link to C.1 – C.4**

<u>Example activity</u>: Introduction to an IPO table

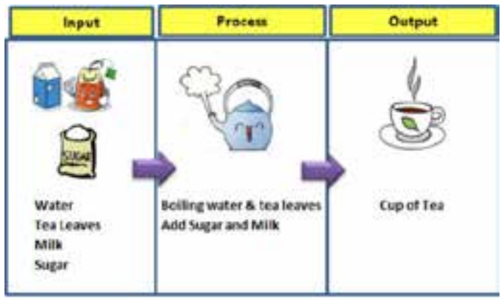
Use a simple algorithm that requires input, processing and output.

Provide each learner with an empty IPO table and a simple algorithm.

Learners need to complete the IPO table, by providing the following:

- Input: What will be the input for their program (e.g., press the 'space' key).
- Process: What actions the program will perform based on the input (e.g., move the sprite up by 10 steps).
- Output: What will be the outcome of the process (e.g., sprite moves up).

Learners then translate the algorithm into code by looking at what the input is (and how the program will receive the input, e.g., ask and answer), what processing needs to be done (and how) and what the output would be (using e.g. say)



Notes/Examples:

Explain Input-Processing-Output (IPO) in a block-based coding example (C.1 – C.4) using a simple IPO tablet.

Learners need to know that.

- An IPO table is a way to organize information about a program's input, process, and output.
- Input: The data or information that is provided to the program at the beginning.
- Process: The actions or operations that the program performs with the input.
- Output: The result or outcome of the process that the program produces.

<table>
<tr><th>Content (Grade 4 / Term 3)</th><th>Notes/Examples</th></tr>
<tr><td></td><td>Elementary IPO table
<table>
<tr><th>Input</th><th>Processing</th><th>Output</th></tr>
<tr><td>What will the input for the program be (e.g., press the 'space' key).</td><td>What actions will the program perform based on the input (e.g., move the sprite 10 steps forward and turns).</td><td>What will the outcome of the process provide (e.g., sprite moves forward and turns).</td></tr>
<tr><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><th>D.8 Interpret a pattern to represent or communicate a message or image</th><td>Link to C.6 and C.7</td></tr>
<tr><td>Example activity
Beavers Anna, Bella and Lena made necklaces to spell out their names.

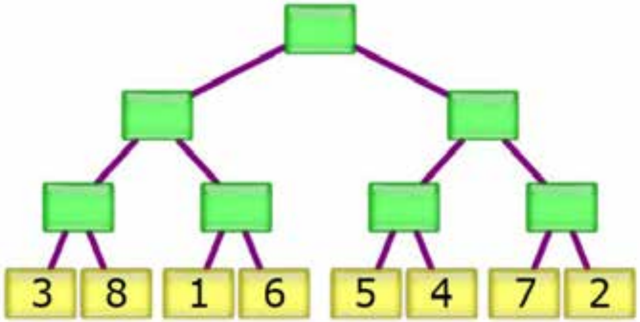
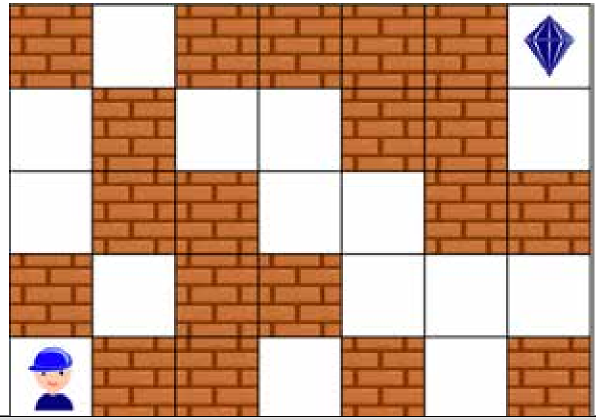
They used different patterns of just two beads for each letter. 

To separate the letters in the necklaces, they used  beads.

The finished necklaces looked like the ones on the right: 

Which necklace did Lena make?
2022-TS-Junior-Question-Paper.pdf (olympiad.org.za)</td><td>Revise the concept of patterns and how they can be used to represent or communicate messages or images.
Show examples of patterns, such as a repeated sequence of shapes, colours, or symbols. Explain that patterns can be used creatively to convey meaning or create visual representations.
Interpret patterns created in C.6 and extend to display on Scratch.</td></tr>
<tr><th>D.10 Demonstrate a basic proficiency in the application of digital skills.</th><th>Link to D.7 and C.1 – C.7</th></tr>
<tr><td>Introduce Paint - Explore the Paint environment.
Open Paint and give a brief tour of the software's interface.
Point out essential elements, such as the drawing canvas, toolbar, colour palette, and different tools available. Demonstrate how to use basic drawing tools, such as the pencil, brush, and eraser. Show learners how to change the size and colour of the drawing tools to create different effects. Encourage learners to experiment with these tools on their own and draw simple shapes or objects.
<u>Example activity:</u> Design a sprite.
Learners design an elementary sprite that they can use as part of their coding, using paint.
Explain saving their sprite as an image file (PNG or JPEG) on their computers using the folder they created for all their projects.
Remind them to choose a descriptive name for their sprite so they can easily find it later.
Importing the Sprite into block-based coding app - learners open the block-based coding environment and create a new project. Guide them how to import their saved sprite image. Learners now create a program using their sprite. Encourage share to share any challenges they faced during the design and import process and what they learned from the experience.</td><td>Learners use Paint to design their own sprites to be used with the block-based app (coding)
Highlight that with their newfound skills, learners can now create more sprites and add them to their coding projects to make them more exciting and interactive.
Reinforce and extend:
open, close and save.
basic file management – learners save their paint files to their working folder.
file naming conventions and extensions (.png. and .jpeg)
navigating to a folder to open a file from within an application to open (or save) a file.</td></tr>
</table>

3.1.4 Term 4

Content (Grade 4 / Term 4)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.2 – C.7 and R.5 – R.7**
Example activity 1 – Abstraction and decomposition Katlego watched a tournament of races and recorded the winners of each stage on the board on the right. The runners wore the same numbers, from 1 to 8, throughout the tournament. Katlego used numbered cards to represent each runner. When the tournament was over his younger brother Tumelo mixed up all the cards, except those from the first stage of the tournament.   TS-2018-Solutions-Guide.pdf (olympiad.org.za) Work out who (which number) the winner of the tournament is. **Example activity 2 Algorithm** The maze shown on the right consists of empty squares and brick walls. John can move from one empty square to the neighbouring empty square horizontally or vertically (not diagonally). John needs to get to the diamond in the top right corner. He has only enough dynamite to remove three (3) walls Use instructions ***Forward, Turn Right, Turn Left and Remove Wall*** to write a set of instructions (an algorithm) for John to get to the diamond by only removing 3 walls. 	Provide small activities for • Abstraction • Decomposition • Pattern recognition (link to C.6 and C.7) • Algorithms When solving problems and creating an algorithm, it is important to notice all the possible conditions (IF construct /branch structures) that depend on the solution of the task. In the tournament task, the condition checking procedure must be repeated (loop construct) until one winner is selected and the problem is solved.
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C.1 – C.5** **C.2 and C3 done together**
C.3 Interpret and execute a given symbolic or written set of commands	

Content (Grade 4 / Term 4)

Example activity 1

The jar must collect as many sweets as possible while walking through the cells in the grid, taking it to the party hat in the top right cell. Each cell in the grid contains 0, 1 2 or 3 sweets.

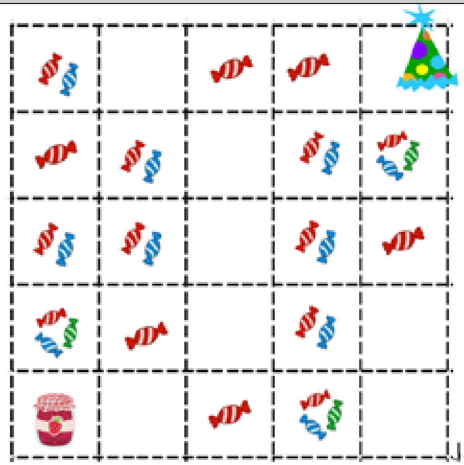
The jar starts in the bottom left cell and must end in the top right cell (with the party hat). The jar can only move upwards and to the right.

Someone wrote the following code and collected x sweets. However, more sweets can be collected.

Change the code to collect the most sweets possible if the jar only moves upwards and right.

In this activity, learners need to interpret code as well as plan code (new route) using computational thinking and change (write) code)

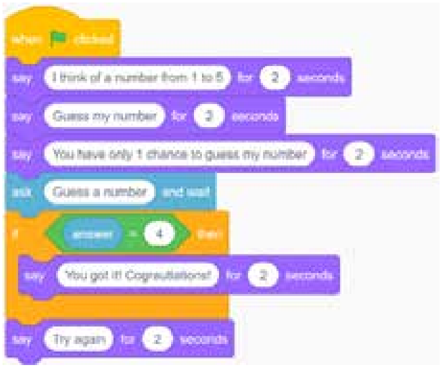
Adapted from: [TS-2018-Solutions-Guide.pdf (olympiad.org.za)]



Example activity 2 Guessing game 1

Guess the number I am thinking of – it is between 1 and 5 – you only have one chance to guess.

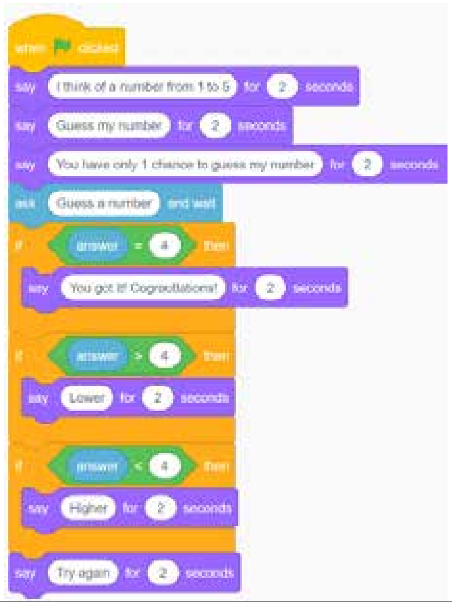
Learners write the code for the above game using the answer block so the player can type in what he/she guesses. (possible solution below)



Example activity 3 Guessing game 2 (possible solution on the right)

Guess the number I am thinking of – it is between 1 and 5 – you only have one chance

Learners improve on the previous activity's code by providing feedback (lower or higher)



Notes/Examples

Learners should write effective code, e.g. use repeat blocks where appropriate instead of repeating actions.

Note:

Provide learners with activities to

- read code and explain what it does.
- work through (trace) / act out code (physically or simulated) to determine the purpose/output or the correctness.

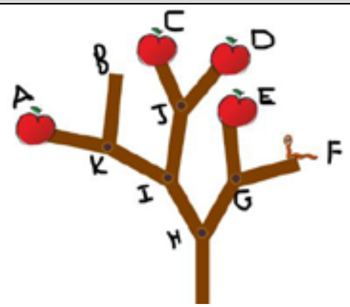
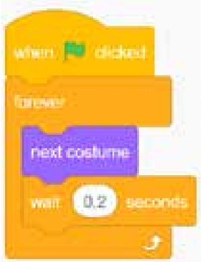
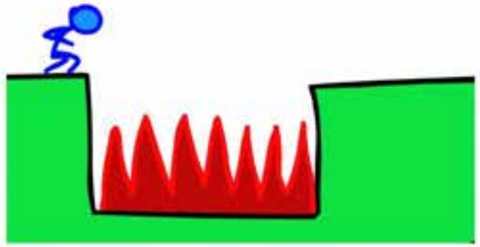
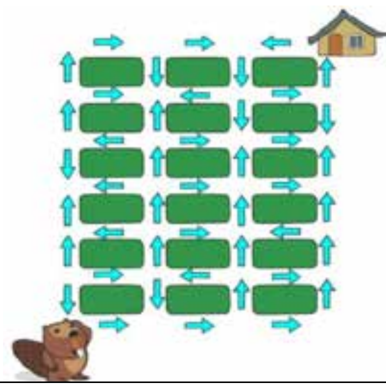
While learners should be able to describe what each line (block) of code does, (describing a code segment line-by-line) it is very important that learners explain the overall purpose of the code, i.e. what the program does/the purpose of the program.

Note:

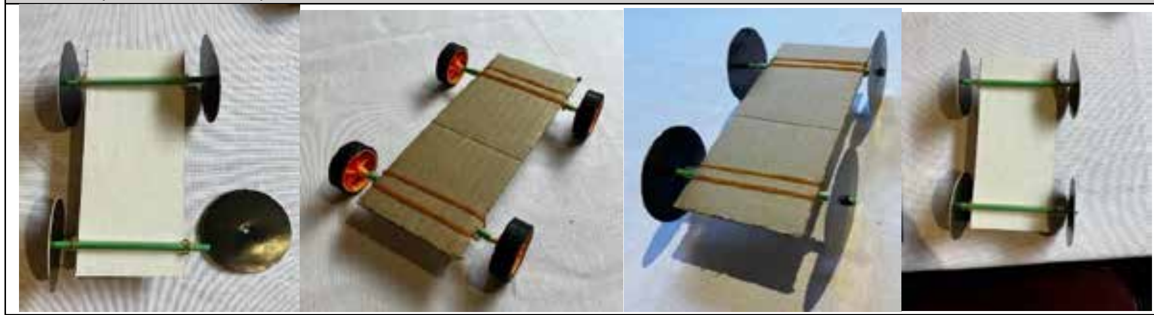
Provide learner with activities enabling them to

- read code and explain what it does or
- work through (trace) / act out code (physically or simulated) to determine the output or the correctness or
- provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or
- translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions))
- add some functionality/instructions to an existing program.
- rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or
- choose the correct solution from 2-3 options or
- compare different solutions to evaluate efficiency or
- debug an algorithm or block-based program (find the bug, describe the bug and correct it)
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

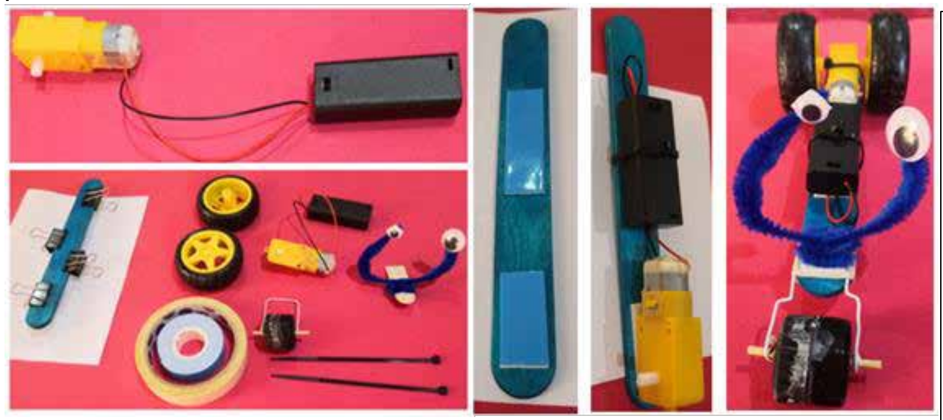
develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

Content (Grade 4 / Term 4)	Notes/Examples
C.4 Debug a given symbolic or written set of instructions	**Link to C.1 – C.3 and C.5 – C.7 and R.6 and R.7**
Example activity 1 Provide learners with code that contains an error, e.g. a logical error (incorrect sequence, incorrect use of control structures, etc.) which they need to find and correct Also, as errors that require debugging are part and parcel of coding, every piece of code they write/problem they solve requires debugging. Teach learners debugging techniques such as tracing code	Debugging is an essential element of programming. One way to debug is to go through your code line by line, reading it and explaining it out loud as you go. This enables you to check the logic in your mind versus what is happening in your code
C.5 Evaluate a given solution towards potential improvement	
Example activity 1 A worm is sitting at the end of the branch (at F) on the tree shown on the right. It wants to eat all the apples by moving through the tree's branches (The tree is made of 1-meter-long branch sections) All the nodes (end nodes (apples/end of branch) and where branches meet)) are named A, B, C…K. • Describe the shortest route using the nodes, e.g. FG→GH→HI→IJ→JC→CJ etc. • In meters, how long is the shortest route? Adapted from [2018-TS-ELEMENTARY-Q-paper.pdf (olympiad.org.za)]  **Example activity 2** Provide learners with a solution to a problem that can be improved, e.g. shortened by using a repeat Also provide learners with two different solutions to a problem and discuss the most efficient/better solution.	Provide learners with • code which they need to improve, e.g. code that repeat which they can shorten by using a repeat (with fixed value) • more than one solution to a problem from which they need to find the shortest/most efficient, etc.
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	
Example activity 1 –Forever, Next costume, jumping Provide learners with the code Let them study the code and explain what it does   **Example activity 2** Provide learners with code where steps are repeated (sequential steps) and they need to identify the pattern and rewrite the code using a repeat with a fixed value (constant)	Provide learners with code where steps are repeated (sequential steps) and they need to identify the pattern and rewrite the code using a repeat with a fixed value (constant)
C.7 Create or complete a pattern to represent a data set	
Example activity 1 In the illustration on the right, each arrow represents one minute of walking. The beaver must follow the arrows; he cannot go in the direction against the arrows. What is the shortest time (in minutes) for the beaver to reach home. Use instructions: Move up Move down Turn left Turn right to create the set of instructions the beaver must follow to meet the conditions of task. [2018-TS-ELEMENTARY-Q-paper.pdf (olympiad.org.za)] 	**Note:** Concrete activities remain important as literature suggests that the primary weakness of today's pedagogy of programming is that it doesn't provide enough opportunity for the novice to develop concrete operational skills, via the correct types of exercises…. due to too much emphasis on writing large amounts of code, and problem solving.

Content (Grade 4 / Term 4)	**Notes/Examples**
Robotics	
R.1 Explain what a robot is in simple terms.	**Link to R.2, R.3, R.4**
R.2 Identify different types of robots.	**R**.1 – R.4 done together Consolidate and revise through retrieval practice. Learners must not be able to • Describe a robot. • Acknowledge the difference and similarities of a virtual and physical robot. • Identify types of robots (physical). • List the various parts that a robot could have. • State what a robot can do (and cannot do). • Describe how robots move. • Explain how robots impact the world around us. Using retrieval practice, this activity encourages learners to actively engage with the material and reinforce their understanding of the concepts related to robots. It also promotes collaboration and critical thinking as learners work together to explore and describe robots in various ways.
R.3 Outline the different components of a robot	
R.4 Present an understanding of how robots affect the world	
Example activity Briefly revise robots with the learners. Recap what they have previously learned about robots and remind them of the main concepts: describing robots, identifying types of robots, explaining their parts, capabilities, movements, and impact on the world. Engage learners in a 'robot explorer' game. Divide learners into small groups of max 4 learners and explain that they or now 'Robot Explorers' on a mission to share information about robots. • Provide each group with a set of flashcards containing questions related to robots (prepared beforehand). This is used to retrieve what the have learned in previous terms. Learners in each group take turns to pick a flashcard, read the question aloud then the group discuss the answer together. Encourage learners to use prior knowledge and memory to answer the questions. • After all the groups have answered their flashcard questions, gather the learners back together. Ask each group to take turns sharing their answers to specific questions. Provide positive feedback and additional explanations as needed to reinforce the concepts. • Now, have each group draw and label a robot on the whiteboard or paper. They should include the different parts of the robot that they learned about, such as sensors, actuators, arms, wheels, etc. Encourage creativity! • After drawing the robot, each group should take turns describing what their robot can do and cannot do based on the concepts they've learned. For example, can it pick up objects, move around, follow commands, etc. • Now, engage the learners in a group discussion about how robots impact the world around us. Discuss both the positive and negative impacts, such as increasing efficiency in industries, performing dangerous tasks, and potential job implications. As a concluding activity, provide learners with a quiz (using apps such as Kahoot!, Google forms, MS Forms, Quizlet, etc.) about what they have learned about robots during the year.	
R.5 Design a simple artefact based on a set of design instructions	**Link to R.1, R.2, R.3, R.5, R.6, R.7 and C.1 – C.7**
R.6 Mimic the operations of a robot	**Note:** R.5, R.6 and R.7 are done together Leaners need to acknowledge the basic principles of wheels and axles and use this knowledge to design and build a simple wheeled robot that can move easily on a flat surface Wheels and Axles: Acknowledge the function and importance of wheels and axles in facilitating movement. Robotics: Basic principles of building a robot, including the integration of wheels and axles. After building the wheeled project, learners start with their end-of-year assessment project.
R.7 Create, test and execute a set of robotic instructions	
Example activity 1 – Desing and build a basic wheeled robot This is an enabling activity before starting with the project – learners will use the knowledge and skills gained with this activity when doing their project. **Steps**  	

<table>
<tr><th>Content (Grade 4 / Term 4)</th><th>Notes/Examples</th></tr>
<tr><td></td><td>Use computational thinking, design thinking and the engineering design process to plan, build, test and debug a robotic artefact – the speedster robot (R.5)</td></tr>
<tr><td>

Example Project (for assessment) Exploring motion with DC motor - example is of a speedster robot connected to a geared DC motor with two points.



Resources required:
- 3v DC motor

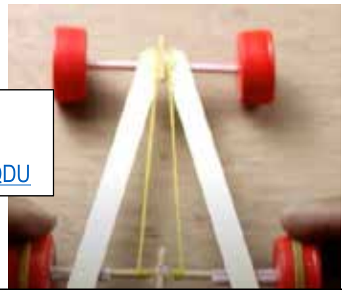
2 x AAA cells / 2 x AA batteries
- AAA / AA battery holder with on/off switch
- Battery box
- Ice cream sticks
- Glue
- Double-sided tape
- Kebab stick
- Bottle caps
- Paper clip
- Sosatie sticks or dowels and straws

Note:
There is no soldering involved.
The battery box wires are connected to the DC motor by twisting the exposed wire to the connectors.
In the example two ice cream sticks are glued together to form a sturdy base. Double sided tape was used to attach the motor and the battery box to the base. The front wheel was made by using a kebab stick and two bottle caps glued together. For "wire" a paper clip was bent to form the front fork.
(There are various approaches that could have been followed to create a front wheel or set of wheels.)

The speedster robot built in R.5 moves, mimicking the operations of a robot.

Alternative
A simpler more toned-down version of a DC car can be created using a simple 3V DC motor, and a fan. For axel casings straws glued and taped to a flat piece of wood could be used. Plastic wheels from a hobby kit with o-rings was attached with cut toothpicks. The example contains no soldering, and the wires were twisted together.

A rubber-band powered car can also be created as shown below.
See: https://www.youtube.com/watch?v=0W9iaB42QDU



</td><td>

Learners need to know
- The working basics of DC motors regarding creating moving objects such as a car
- The principles of DC motors and how it is used to drive motion in objects

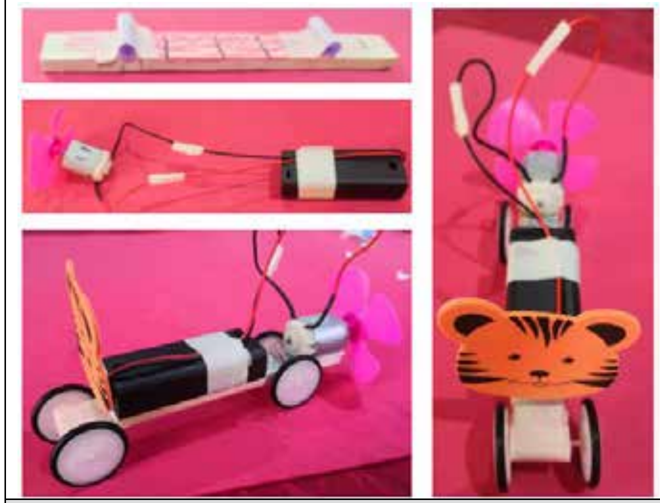
Link to learner's prior knowledge of wheels an axle.

This is done with
- R.7 (design, test and execute and debug a set of robotic instructions) and
- R.6 – when the artefact moves, it illustrates the operations of a robot

The design of the "speedster" robot was done by a grade 4 learner under the guidance of an educator. The learner suggested the use of cable ties.

As part of the first construction the learner discovered that the motor was placed "upside-down" having the speedster go backwards. As part of the problem solving and improvement process the learner figured out that he had to flip the motor. Before the final reassembly the rotation of the wheels was checked. The final speedster then correctly drove forward when switched on.

</td></tr>
</table>

Content (Grade 4 / Term 4)	Notes/Examples
Alternative 	**How motors are used to drive robots.** Motors in robots make them move. They use electricity to spin and create motion. Robots have motors placed in different parts, like wheels or arms. If a robot has wheels, the motor makes them turn. If it has an arm, the motor controls its movement. The motors are controlled by signals that tell them how to move, like going forward or lifting an arm. **Note** When learners design and build robotics artefacts, they use computational thinking and design thinking and follow the engineering design thinking process which include executing, testing and debugging.
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.3 and D.7**
Example activity 1 As a concluding activity, provide learners with a quiz that test their knowledge about what they have learned so far. **Include pictures, diagrams, etc. as part of the quiz**	**Retrieval practice** Create a quiz (using apps such as Kahoot! Google forms, MS Forms, Quizlet, etc.) about what they have learned so far
D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.6 and D.7**
Digital Footprint – Does what you do online always stay online? Learners learn that the information they share online leaves a digital footprint or "trail." Depending on how they manage it, this trail can be big or small, and harmful or helpful. Learners compare different trials and think critically about what kinds of information they want to leave behind. Discuss how these footprints can be permanent and visible to others, potentially affecting their reputation, relationships, and future opportunities. People can also use this information to harm you, e.g., it can lead to stealing your identity, malicious individuals stalking you online, tracking your physical location, etc.  **<u>Example activity:</u>** Collecting information and discussing privacy. Create a handout with questions that learners should answer. Provide space for three names. E.g., What is your name? What is your surname? How old are you? What is your birthdate? What is your favourite colour? How many siblings do you have? Tell learners to gather data about three other learners. Have a discussion in class about the data gathered and why certain information should not be shared.	In the online/digital world, we all leave a digital footprint (just as the one we leave when walking on sand). This digital footprint links all online activities on the internet like visiting websites, posting, liking, commenting, etc. to a person.
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.1**
Reinforce and extend from the previous terms. The concepts of "hardware" and "software". Use simple and relatable examples to explain the difference between the two. Show some real hardware components, such as a keyboard, mouse, and USB drive, and describe their functions. Explain that hardware refers to physical parts of a tech device, while software consists of programs and instructions that make the hardware work.	

Content (Grade 4 / Term 4)	Notes/Examples
D.4 Identify the common uses of ICT in the real world	**Link to D.5**
D.5 Differentiate between the components of an ICT system	**Link to D.4**
Example activity 1 As a concluding activity, provide learners with a quiz that test their knowledge about what they have learned so far. **Include pictures, diagrams, etc. as part of the quiz**	**Retrieval practice** Create a quiz (using apps such as Kahoot!, Google forms, MS Forms, Quizlet, etc..) about what they have learned so far
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to all D.1 to D.5, D.7 and D.10**
Example activity: Positive and negative impact of ICT In small groups, learners discuss some positive and negative impacts of ICT and make a list of two positive and two negative impacts. Learners now need to suggest solutions to minimize the negative impacts and to make the best use of technology. For example, answering the following questions: • How can we limit our screen time and balance it with other activities that enrich our lives, such as also spending time with family and friends? • How can use technology mindfully and purposefully, rather than impulsively and compulsively. • How can we protect our privacy by using strong passwords? • How can we ensure a positive footprint? • How can we treat others with respect when online? Groups create a poster that highlights the positive and negative impacts and provide solutions for minimising negative impacts/	Revise and extend from Term 3 using different activities. Learners need to acknowledge that ICT impacted our lives / the world we live in, both positively and negatively and have its drawbacks and challenges. **Positive** • Enabled us to communicate, learn, work, and entertain ourselves more easily and efficiently. **Negative** • Can be addictive, distracting, and isolating which could affect our mental and physical well-being, as well as our interpersonal relationships and social skills. • Can pose risks to our privacy, security, and environment. • Can create social inequalities, ethical dilemmas, and cultural conflicts. Technology can also affect our mental and physical well-being, as well as our interpersonal relationships and social skills.
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.7 and C.2 – C.5**

Example activity: Revise and extend the concept of input, processing and output.
Teacher demonstrates input (e.g., typing into a search engine such as Google and output (the results (webpages) of the search displayed on the screen) and explains that the input was processed (looking for the information on the internet) to provide the results.
In pairs, learners need to write down examples of input, and the resultant output, then describe what processing they think took place to provide the output, e.g. typing a text message on a keyboard (WhatsApp) to be sent to a friend or clicking on icons or buttons on a GUI to perform certain actions such as opening an application. Processing - when you perform a search on a search engine, the system processes your query to find relevant results from its index.
Output - The search engine displays a list of relevant web pages as search results.
	Reinforce and extend from D.7 Term 3
	Learners need to acknowledge that:
	• Input, processing, and output are fundamental concepts in computing and information processing.
	• They are essential components of any system that deals with data and information.
	• Users provide data as input, which is processed by the computer system to produce meaningful results as output.
	  ○ Input refers to the data or information that is provided to a computer system for processing.
	  ○ This data can come from various sources, such as keyboards, mice, touchscreens, sensors,
	• Processing refers to the manipulation and transformation of the input data by the computer.
	  ○ The processing step involves various operations, such as calculations.
	• Output is the result or information produced by the computer system after processing the input data.
	  ○ Output can be displayed in the form of text, graphics, images, etc. on a computer screen or a printer or sound through a speaker

D.8 Interpret a pattern to represent or communicate a message or image	**Link to D.9 and C.1**
D.9 Create a pattern to represent or communicate a message or image	**Link to D.8 and C.1**

Content (Grade 4 / Term 4)	Notes/Examples
Example activity – Create a cipher disc Pair the learners. Provide each pair with two paper plates, scissors and a pencil. Cut the inner disk from the centre of a paper plate. Write the letters of the alphabet around a whole paper plate. Make sure they are equal distances apart. Place the inner disk on top of the whole plate. Randomly write letters of the alphabet on the inner ring. Make sure that each letter lines up with a letter on the whole plate.     Once done, each pair write their own messages and write cryptograms. Then each pair tries to solve the encrypted message of another pair.	At a basic level, learners need to know that: Encryption is linked to cybersecurity and is a way of **protecting** the **confidentiality** of messages by making them **unreadable** to anyone who does not have the **key** to decode/ decrypt them Cybersecurity is the act of keeping information, ranging from credit card numbers to national secrets, private and viewed by only the right people. Encryption and decryption go together. D.9 – Learners encrypt a message using a cipher D.8 – Learners interpret the cipher and decrypt the message.
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link D.8 and D.9.**
Guide learners through the process of creating simple repetitive patterns using Microsoft Paint. Emphasise that duplication is used to form patterns. Encourage learners to experiment with different colours and sizes to create their own unique patterns. Have learners create their own digital artwork using patterns they have learned or discovered in Paint. Learners also create sprites and backgrounds to import for their coding apps.	Done in relation to C.2 – C.7 Emphasise good • file management • File naming conventions • File extensions (.sb3 and .png /.jpeg)

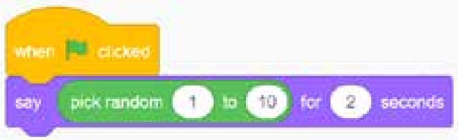
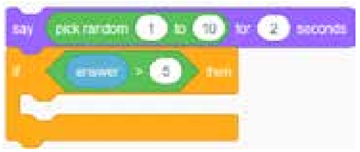
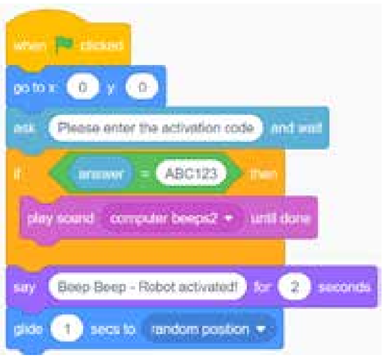
3.2 GRADE 5

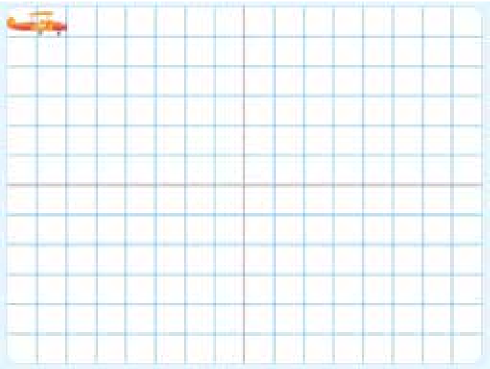
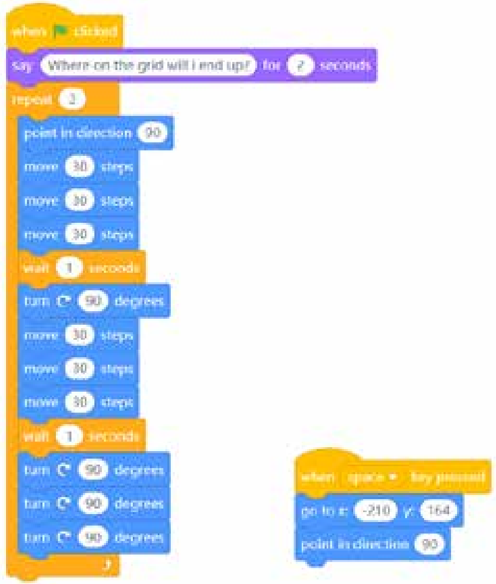
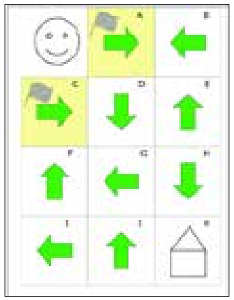
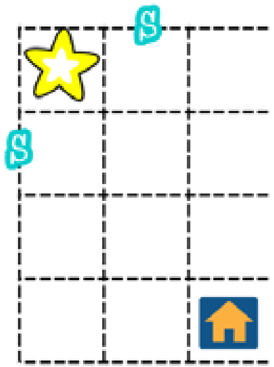
Note:

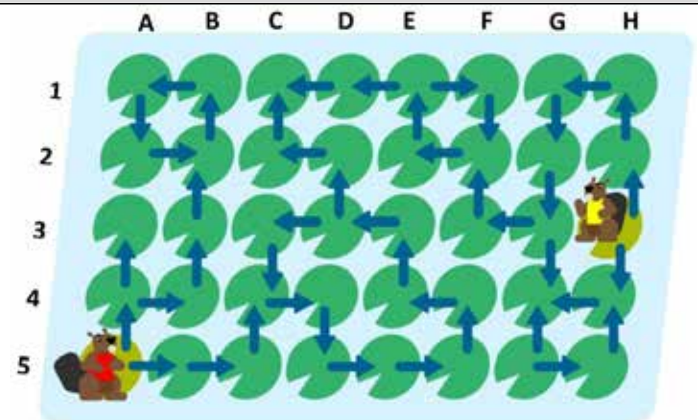
Teachers must include the following competencies and content in their Annual Teaching Plan (ATP), distributed across the terms and sequenced, organised and grouped in a manner that will facilitate learning in a manner that will make sense for learning and teaching, maximize the learners' learning outcomes and achievement. and in a way that will make optimal use of time and resources. Some competencies could also be combined in bigger/more complex programs.

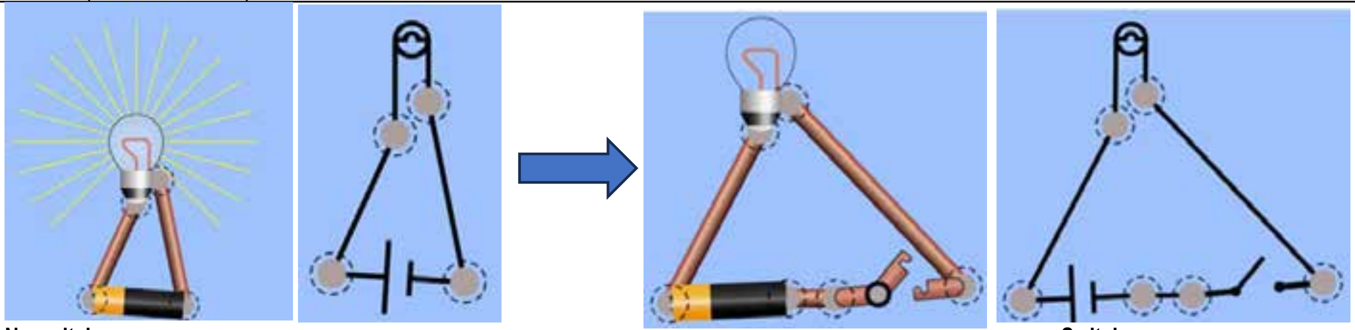
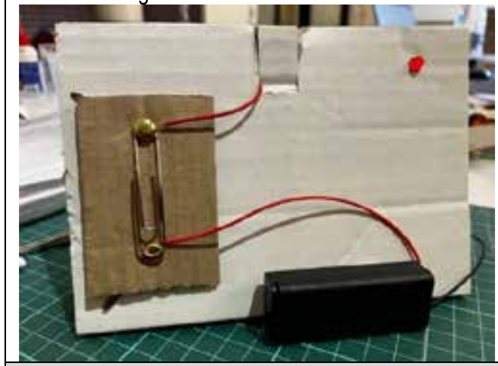
Also, in Grade 5, it is advisable to complete all the Coding in Term 1 and Term 2 and then complete the Robotics in Term 3 and Term 4. Digital concepts are spread across the four terms as it supports both Coding and Robotics.

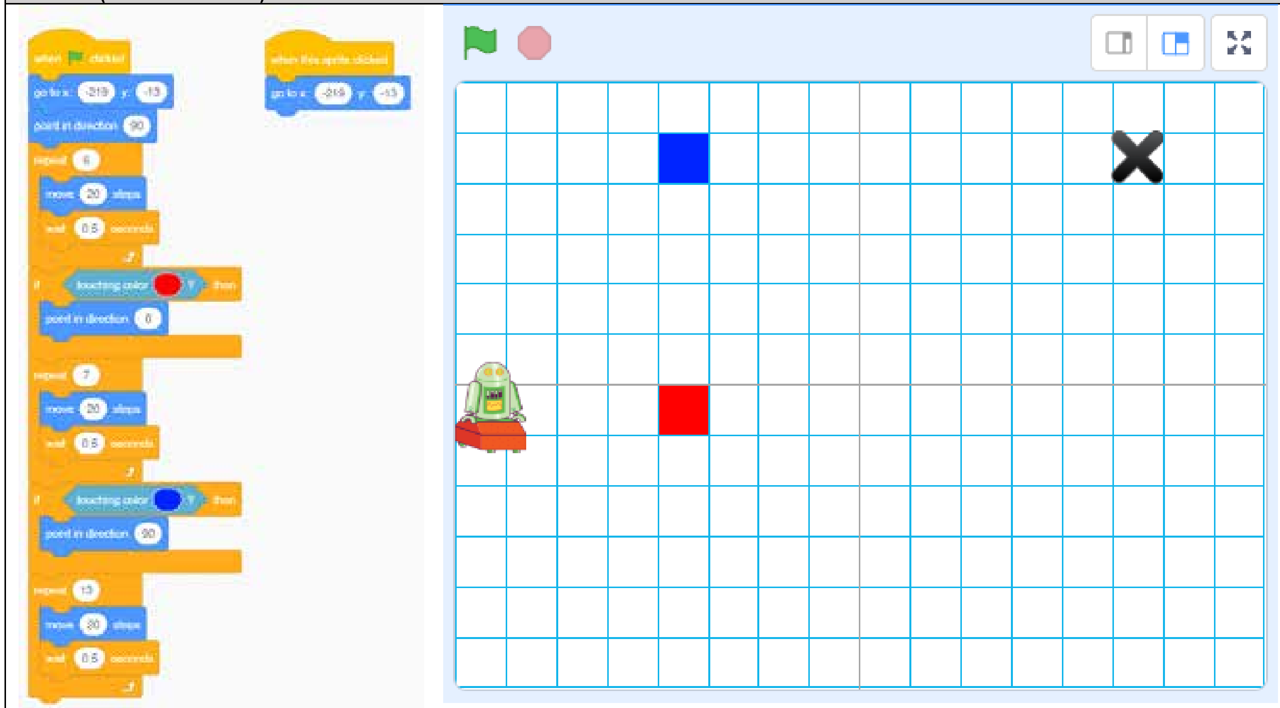
3.2.1 Term 1

Content (Grade 5 / Term 1)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.1 – C.7, R.5 – R.7**
Computational thinking is applied in all coding activities	Computational thinking is used when solving coding problems
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C1, C.3 – C.7 and R.5 – R.7**
Example activity 1 – Introduce *pick random number* Provide learners with the code on the right. Learners run the code several times and each time write down the output (the number displayed) Learners then explain what the code does.  **Example activity 3 – Complete code (missing code instructions)** Complete the code on the right as follows: If the number displayed is greater than 5, the sprite must say "greater than 5". All the code must be executed 5 times.  **Example activity 4 – Algorithm to code, implement test and debug (introduce *glide to random position*)** A robot can only respond if the correct activation code is entered. Code the following algorithm to implement in a block-based coding environment. 1. Robot starts at position x:0;y:0 2. Ask the user to enter the activation code. 3. If the activation code is correct (ABC123) the computer plays a beep sound 4. The robot responds by saying "beep, beep Robot activated!" for 2 seconds. 5. The robot then glides to a random position. **Possible solution**  **Example activity 5 – Debug code** Provide learners with a problem and incorrect code to solve the problem and let them debug the code to solve the problem. **Example activity 6 – Open ended** Learners use their knowledge, skills and experience to design, code, implement, test and debug a program of their choice	**Note:** Provide learner with activities enabling them to • read code and explain what it does or • work through (trace) / act out code (physically or simulated) to determine the output or the correctness or • provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or • translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions)) • add some functionality/instructions to an existing program. • rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or • choose the correct solution from 2-3 options or • compare different solutions to evaluate efficiency or • debug an algorithm or block-based program (find the bug, describe the bug and correct it) • develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging. develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

Content (Grade 5 / Term 1)	Notes/Examples
C.3 Interpret and execute a given symbolic or written set of commands	**Link to C.1, C4, and C.6**
Example activity In the following example the learners must interpret the code and determine where the aeroplane will end up on the stage. Each small block in the grid is 30 x 30, implying a move of 30 steps will move the aeroplane one block forward.  Alternative solution  	The learners must interpret the code and determine where the aeroplane will be once all the code has executed. **Note:** While learners should be able to describe what each line (block) of code does, (describing a code segment line-by-line) it is very important that learners explain the overall purpose of the code, i.e. what the program does/the purpose of the program. **Note** **An Algorithm** is a set of well-defined steps or instructions that are followed to perform a specific task or solve a particular problem. The instruction set can be sequential or can include branching (decision structure) or repetition (loops). **Key characteristics of a good algorithm: Each step** • must be clear and unambiguous. • must be at the right level of ***detail*** and specific. • consists of a single task (be at the most basic level) • must be in the correct, logical ***sequence*** • must be correct/solve the problem
C.4 Debug a given symbolic or written set of instructions	**C.3 and C.4 done together and C.6**
Example activity 1 – Debug Help Smiley get through the arrow maze back to his house. When standing in a square with an arrow, Smiley must move to the next square by following the direction of the arrow. Smiley can choose to start from either shaded square with a start flag. Now it is impossible for Smiley to reach his house. By changing the direction of one of the arrows, Smiley will be able to follow the arrow to his home. Which arrow needs to change direction?  **Example activity 2** Use the corrected algorithm from the above activity and write the code in a block-based environment for Smiley, the star to get home. (The background is customised, imported into a block-based coding environment) The Ss indicate the starting points 	C.3 is generally done with all programs that learners write, complete or change **Note:** It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively, using different activities and combinations of concepts.

Content (Grade 5 / Term 1)	Notes/Examples
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to C.1 – C.4 and R.5 and R.6 and D.6 and D.7**
Example activity On the lake, beavers can go from one lily pad to another but only in the direction the arrows indicate. Bob (in a red vest) and Nora (in a yellow vest) would like to meet on one of the lily pads. They start on different lily pads as you can see below. Which lily would they meet on? Pack out (using the arrows) the path that each must follow to be able to meet.  	It is important that pen-and-paper coding exercises are not neglected. By identifying patterns, we can predict what will come next and what will happen again and again in the same way. A pattern may be numerical, visual or behavioural. In Computer Science/coding one analyses patterns in data and make predictions and generalisations based on the pattern analysis.
Robotics	
R.1 Explain what a robot is in simple terms. **R.2 Identify different types of robots.** A robot is also an artificial agent, meaning it acts as a substitute for a person, doing things it is designed for. The term comes from a Slavic root, robot-, with meanings associated with labour. Robots are usually machines controlled by a computer program or electronic circuitry. They may be directly controlled by humans. They may be designed to look like humans, in which case their behaviour may suggest intelligence or thought. Most robots do a specific job, and they do not look like humans. They can come in many forms. **Different types of robots:** **Mobile robots** are designed to move and navigate in different environments. • They can have wheels, tracks, or legs to enable mobility. • Examples include robot vacuums, delivery robots, or rovers used for planetary exploration. **Industrial Robots**: used in manufacturing and production processes. • They are typically large and powerful machines. • Designed to perform repetitive tasks, such as assembly, welding, or packaging in factories. **Autonomous Robots**: Autonomous robots are capable of operating and making decisions without constant human control. • They have onboard sensors, processors, and algorithms to perceive and navigate their environment. • Can perform tasks and adapt to changing conditions independently. **Remote Controlled Robots**: Operated by a person through a remote-control device • Remote controlled robots are operated by a person through a remote-control device. • They require constant human input and control. • Movements and actions are controlled by a person from a distance, using joysticks or buttons on the remote control.	R.1 and R.2 can be done together Reinforce and extend from Grade 4: Include concept of artificial agent and the origin of the term 'robot' Discuss types of robots: Mobile, Industrial, autonomous, remote controlled
R.3 Outline the different components of a robot	**Link to R.5**
Example activity Present the basic concepts and principles of electric circuits and switches and how electricity flows from the energy source, through the conductor and the load. Basic electrical components: Knowledge of batteries as a source of energy, wires as conductors, and a bulb as a load in the circuit. Basic flow of electricity: Know how electricity flows from the energy source, through the conductor, and to the load. Basic switches: Know the function of a switch in a circuit, which is to control the flow of electricity.	Learners need to know that, when a robot uses a power source such as batteries, it includes basic electrical components such as circuits and switches. Therefore, they need to, at a basic level, know the basic concepts and principles of the above. Learners need to know how

Content (Grade 5 / Term 1)	Notes/Examples
 No switch Switch	• energy can be stored in cells or batteries, • a circuit transfers electrical energy, and • how a switch can control the flow of electricity
R.5 Design a simple artefact based on a set of design instructions	**Link to R.3**
Learners design and build their own basic electric circuit to light a bulb 	Learners build a simple circuit with a power source, light bulb and switch Learners must be able to, at an elementary level, describe a circuit and how it works.
R.6 Mimic the operations of a robot	**Link to R.5 and C.1-C.7**
Example activity Design a grid with obstacles and a virtual robot (sprite). The sprite moves on the grid and when the sprite touches an obstacle, it responds in a specific way.	Learners mimic the operations of a robot using a virtual robot in the block-based coding environment. Use a grid background with obstacles placed on the grid. The sprite/virtual robot performs specific actions when it touches an object. In terms of problems that provide a partial solution where some code instructions are missing and learners must fill in the missing code instructions, the concept of Parsons Puzzles could be helpful as it provides scaffolding for learning programming. It helps learners to develop logical thinking, Refer to Grade 4 Term 1 C.3.

Content (Grade 5 / Term 1)	Notes/Examples
	**Note**: Evidence suggests that pupils should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practice (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).
R.7 Create, test and execute a set of robotic instructions	**Link to R.5 and C.1 – C.7**
Example activity Using computational thinking, design thinking and the engineering design process: • Plan, create a basic electric circuit with a switch to turn on a light bulb. • Execute and test the design and debug/fix if required.	Do with R.5 Learners need to use computational thinking and design thinking while following the steps on the left when they design an artefact based on a set of design instructions (R.5)
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.3, and D.7**
Learners should be able to: • explain what a computer in the context of information technology is • relate the concept of computers to that of an IT tool **Example activity:** Explain the basic concept of computers in everyday life Explain the basic concept of computers in the context of information technology. Define what a computer is, emphasising that it is an electronic device capable of performing various tasks using instructions. Present examples of IT tools on presentation slides or posters (e.g., laptops, smartphones, tablets, servers, etc.). Discuss how each of these devices is a type of computer that serves different purposes within the field of information technology. Display images of various computers commonly used in everyday life, such as personal computers, smartphones, ATMs, self-checkout machines, etc. Ask learners to list the examples they recognise and relate the use and purpose of each computer to their own daily routines and activities.	**D.1, D.3 and D.7 can be done together** Learners need to understand that Information Technology (IT) specifically refers to the use of computers, networks, and software to manage and process data and information. The purpose of Information Technology (IT) is to use computers, software, networks, and other technology tools to manage, process, store, and present information in various contexts. A computer is an electronic device that processes data and performs various tasks according to a set of instructions provided through software or programs.
D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.4 and D.6**
Give a basic explanation of the digital world all around us. Provide a basic description of a digital world and digital citizenship.	**D.2, D.4 and D.6 can be done together**

Content (Grade 5 / Term 1)	Notes/Examples
Discuss how digital citizenship contributes to a positive digital world. Include caring for your device as a good citizen. **Example activity:** Exploring the digital world like a magical playground. Explain to learners that the digital world is like a big, magical playground that exists inside computers and the internet. It's a place where you can do so many cool things, just like in the real world, but even more exciting and full of possibilities! Let learners create a diagram in Paint where a magical online playground is simulated. Ask them to include shapes/patterns that represent communicating with friends, exploring the world, sharing pictures with family, etc. Discuss that like any playground, the digital world comes with some rules to keep everyone safe and happy. Being kind to others is essential, just like you would in the real world. Avoid using mean words or doing things that might hurt someone's feelings. Treat people online the way you'd want them to treat you. It's also essential to be careful about sharing personal information with people you don't know well. Always check with your parents or teachers before giving out any personal details, like your full name, age, address, or school. Continue with the activity and ask learners to include symbols of the good digital citizenship road signs (see Grade 4 Term 2 D.2) created about rules to follow, to be a good digital citizen.	The digital world is an interconnected realm of information and communication technology that surrounds us in our everyday lives. It encompasses various digital technologies and platforms that facilitate the creation, processing, storage, and sharing of data and information through electronic means. The digital world has transformed the way we live, work, communicate, and access knowledge. Digital citizenship refers to the responsible, ethical, and respectful use of digital technologies and the internet. It involves understanding the rights, responsibilities, and risks associated with participating in the digital world. Digital citizenship encompasses various aspects, including online behaviour, information literacy, digital communication, privacy, security, and copyright awareness. Being a responsible digital citizen means using technology responsibly, treating others with respect online, protecting personal information, and being mindful of the impact of one's actions on others in the digital space. By embracing the principles of digital citizenship, you become a positive force in the digital world. You contribute to creating a safe, respectful, and supportive online environment for everyone to enjoy and learn. Just like in the real world, being a good digital citizen is a lifelong journey of learning and growth.
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.1, R.3, R.**
Revise and extend the concepts of "hardware" and "software". Show some real hardware components, such as a keyboard, mouse, and USB drive, and describe their functions. Discuss the concept of a computing device, what it is and its importance in our daily lives. Choose a device and observe and examine the external components of a specific computing device. Ask them to identify and describe each component, such as the screen, keyboard, touchpad/trackpad. Discuss the basic function of each. Identify the apps found on these devices, e.g., block-based coding app. (Link to D.1. and D.2.) • Provide a basic description of a computing device, including the concepts of input, processing, output, and storage. • Distinguish between the concepts of hardware and software. • Provide a list of common computing devices and describe what they are used for. • Provide a list of common apps found on devices (e.g., WhatsApp) (Link to D.1 and D.2) • Describe and demonstrate the concept of working in and navigating an application (app). Link to C.2) • Navigating applications • Allow the learners to locate and open the different applications on their devices. • Request the learners to write down the common components in the different user interfaces of the applications they are working with • Discuss the purpose and functions of common buttons and icons that learners encountered • Request the learners to write down the different components in the user interface of the applications • Discuss the purpose and functions of new/unknown buttons and icons that learners encountered	Learners need to: • Know what a computing device is (electronic machines capable of processing data and performing tasks according to instructions.) • Distinguish between hardware and software, • Name common computing devices and their uses • Name common apps found on these devices, • Know that a computing device is made up of hardware and software • Know that hardware refers to physical parts of a tech device, while software consists of programs and instructions that make the hardware work. **Note:** Some aspects covered here can be done while learners work on computing devices in class, e.g., while busy with Coding and Robotics
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.2 and D.3**
Reinforce and extend from the previous grade.	Focus on the evolution from technology (T) to information technology (IT) to information and communication technology (ICT) Remind learners"

Content (Grade 5 / Term 1)

Remind learners that technology is all around us (and has been for centuries), and it includes all the tools and machines we use to make our lives easier and better. It can be something simple, like a pencil or a bicycle, or something more complex, like a computer or a smartphone. Technology helps us do things faster, communicate with others, and learn new things.

If the teacher has access to old technology devices (e.g., old cell phones, a floppy disk, a VCR, typewriter, and telephones), bring the artefacts to the class. Allow learners to handle and explore these older devices, comparing them to the current technology they use. Alternative show the learner's pictures of old artefacts.

Example activity: Impact of technology – Tech Time Capsule

Explain the concept of a time capsule as a container that holds items representing a specific time for future generations to discover.

Discuss with the learners the idea of creating a "Tech Time Capsule" to capture the impact of technology in their lives right now. Distribute worksheets or paper to each student and ask them to reflect on the impact of technology in their lives. (Link with other subjects such as NST, SS, LS). Prompt questions like:

- How has technology changed the way you learn or play?
- What are some of your favourite technological gadgets or apps, and why do you like them?
- How has technology changed the way your family communicates or spends time together?
- Can you think of any ways technology has helped people in the world, such as in emergencies or healthcare?
- Now ask learners to draw a picture of a device and place it in the "Tech Time Capsule". Let them fold the pictures up and place them in a box.

Notes/Examples

Technology can be anything that makes our lives easier (e.g., electricity).

Information Technology (IT) is a special kind of technology that focuses on computers and how we use them to process and manage information. IT includes things like computers, laptops, tablets, and the software we use to create documents, play games, and do many other things. IT helps us store and organize information, like pictures, videos, and documents, so we can access them whenever we need them.

Information and Communication Technology (ICT): Add one more concept. Communication. Information and Communications Technology, or ICT, is a big idea that combines Information Technology with how we communicate with others. It's like bringing together computers and other devices, like smartphones and the internet, to help us talk to our friends and family, even if they are far away.

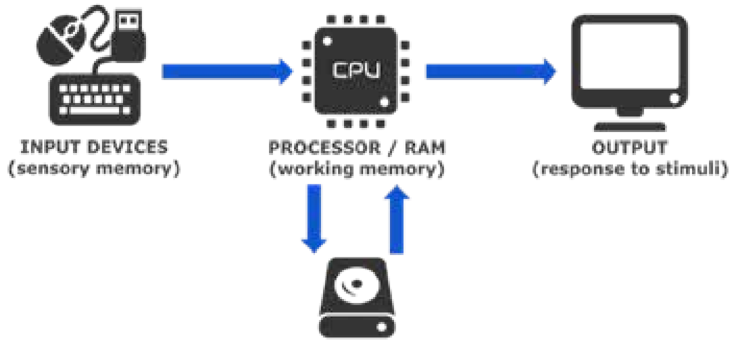
D.7 Present a basic understanding of the concept of input processing and output.

Provide a basic description of a computing device, including the concepts of input, processing, output, and storage. Let learners draw a diagram for understanding. Connect the diagram to the IPO table.

Revise the use of the IPO table. Give some real-life examples of the IPO process.

Connect to the IPO cycle in a computer.

Explain the concept of GIGO to the participants. Tell them that GIGO is an important principle in computer science and data analysis, which emphasises that the quality of output is only as good as the quality of the input data.



<u>Example activity</u> GIGO story telling.

Divide the learners into small groups of 3-4 people. Each group will create a short fictional story collaboratively. However, there's a twist: the story must include some absurd or ridiculous elements that don't make logical sense. Emphasise that the story should be funny and creative but shouldn't follow any coherent structure. After the groups have finished creating their stories, have them share their stories with the rest of the class. As each group shares their story, ask the others in the class to identify the absurd or illogical elements in the story.

Lead a short discussion about how the concept of GIGO applies to the activity.

Summarise the activity by reinforcing the importance of data quality and how it affects the results we get from any process and computer algorithm.

Link to C.1 – C.5

Elementary IPO table

Input	Processing	Output
What will the input for the program be (e.g., press the 'space' key).	What actions will the program perform based on the input (e.g., move the sprite 10 steps forward and turns).	What will the outcome of the process provide (e.g., sprite moves forward and turns).

Everyday examples

Content (Grade 5 / Term 1)	Notes/Examples
D.8 Interpret a pattern to represent or communicate a message or image	**Link to D.9 and C.1**
Example activity: **Plotting a map using patterns**. • Print out the map and ask learners the following questions. • Give learners a route e.g., 1, 5, 16, 15, 11, 7, 18, 4, 19, 10, 14 and ask them to trace the route with a marker. Give several routes to follow and use different colours. • You are a game ranger at the Kruger National Park, and you must work out a one-day route that takes tourists to main attractions. Follow the following rules: • Tourists will start at Orpen and want to make a half-way stop at Satara. • Tourists may not pass through the same point twice (where possible). • They must start and end at the same place. • The tourists would like to visit as many points as possible. • Write down the route you suggest by giving the number sequence.  	• Guide the learners through the process of creating simple repetitive patterns. Link to patterns created in C.6. and C.7. • Teacher can include different questions based on the context of the learners. Include map of the area where learners stay. Ask similar questions.
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.1 – C.7 and R.5 – R.7**
Reinforce and extend from the previous year. **<u>Example Activity</u>** File and folder management Activity done in relation to prior knowledge and skills in previous grades. Explain the importance of file and folder management on a computer. Discuss how organizing files and folders helps in easy access to information, reduces clutter, and improves productivity. Define what files and folders are and explain their relationship (folders can contain files or other folders). Show examples of common file types (e.g., documents, images, videos) and folders (e.g., My Documents, Pictures, Videos). Demonstrate how to create a new folder. Have learners follow along and create their own folders with different names. Discuss best practices for naming folders (e.g., using clear and descriptive names).	Revise open, save, close, etc. Basic file management – create own folder. Open and save from own folder. File names

3.2.2 Term 2

Content (Grade 5 / Term 2)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.1 – C.7**
See example activity below	Computational thinking is the foundation of all programming tasks. Provide pen-and-paper computational thinking activities to develop computational thinking.

Example activity

You have two containers – an empty container and a container with milk. You also have a glass filled with orange juice. You want to pour the milk into the glass to drink, but you also want to save the orange juice for later.

Two different people compiled instructions to achieve the task using the available objects only: the two containers and the one glass filled with orange juice. However, the instructions do not seem to work properly.

Study the different instructions. None of the instructions are working correctly.

Instruction set 1	Instruction set 2
1. Take off the cap from the milk container	1. Take off the cap from the empty container
2. Pour the milk into the empty container	2. Pour the orange juice into the empty container
3. Put the cap back on the container	3. Pour the milk into the glass
4. Pour the orange juice into the milk container	4. Put the cap back on the now empty container
5. Put the cap onto the container now containing the orange juice	5. Rinse the glass
6. Rinse the glass	6. Rinse the now empty container
7. Pour the milk into the glass	7. Put the container with the orange juice back in the fridge
8. Drink the milk	8. Put the cap on orange juice container
9. Put the container with the orange juice in the fridge	9. Drink the milk

Use the instructions and write you own set of precise instructions so that the task can be achieved in the most efficient (the shortest and best way without mixing the milk and orange juice in any way) manner and be understood by anyone following it.

Does one need an extra container?

Content	Notes/Examples
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C.1 C.3 – C.7**
See example activities below	Provide learners with pen-an-paper activities to practise the Join instruction. Token based tangible coding applications can also be deployed to strengthen the mastery of content and outcomes.

Example activity 1 – Introduce *join block (two aspects only, e.g. say with text and answer)*

Provide learners with the following code which they need to study and explain what it does.

Let them run the code to check their interpretation and ask them to specifically explain what the ***join*** instruction does. Explain the join block if necessary. Then, on a worksheet, provide various instructions that they need to join and which they must test in a block-based coding environment.

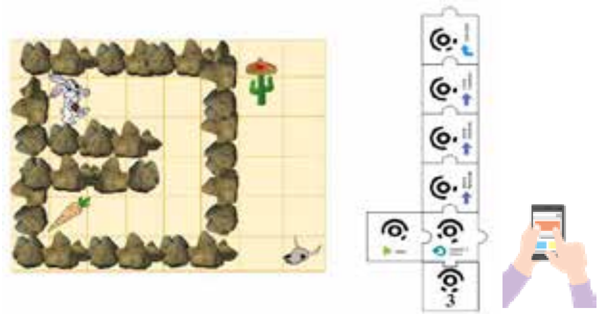
Example activity 2 – Introduce Point towards mouse pointer

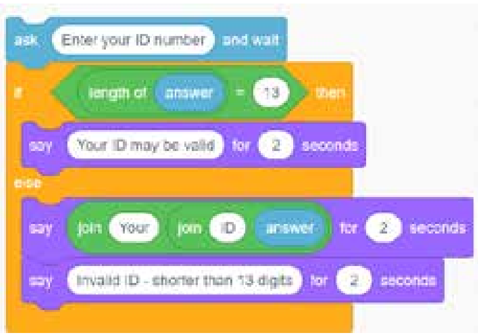
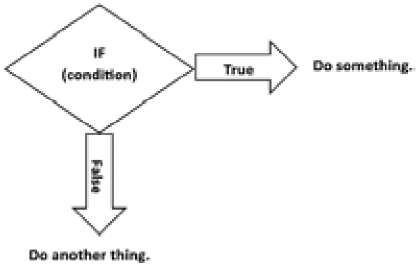
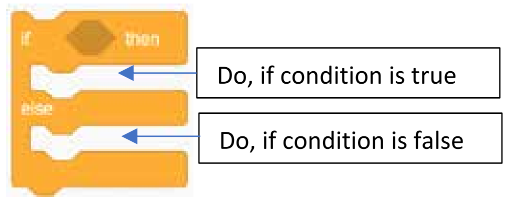
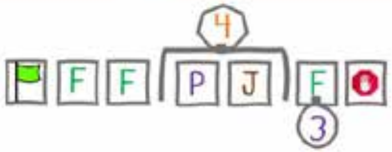
Run the code on the right (keep moving your mouse on the stage)

Explain what the code does.

See if you could have the beetle draw a triangle based on

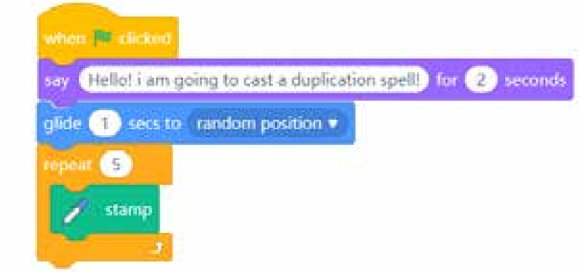
How you move your mouse.

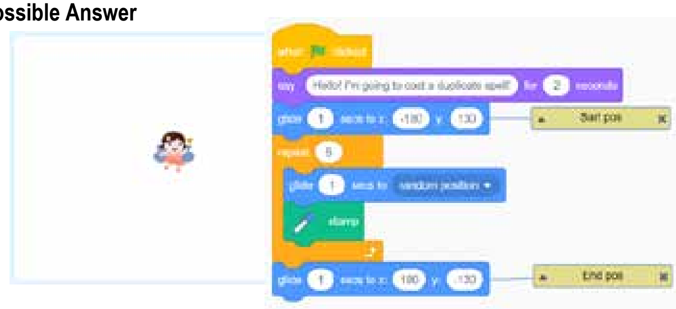


Content (Grade 5 / Term 2)	Notes/Examples
C.3 Interpret and execute a given symbolic or written set of commands	**Link toC.4**
Example activity 1 – Introduce ***length block*** and ***stacked join*** blocks. Provide learners with the code on the right. Learners study the code and explain what it does. Learners run the code and compare with what they expected. Let learners explain what the length function does. Let them come up with ideas where it could be useful when programming. Learners now write their own program using the length function.   Explain: →what a **condition** is →branching (flow)	 Learners must be able to explain ➔ What a condition is ➔ The branching or the flow (based on the condition) ➔ The difference between an if…then… and an If…then…else ➔ Understand when two separates if…then…statements can be written as an if…then…else statement Provide learners with pen-an-paper activities to practise the Join instruction as learners generally struggle with the layers (stacks) when using more than two 'joins'
C.4 Debug a given symbolic or written set of instructions	**Link to C.3**
Example activity 1 The codes F, P and J, on the right represents the instructions Forward, Pick-up and Jump respectively. The numbers indicate how many times an instruction must be executed. SSB, wants to pick up all the carrots and land on the X block at the end. The code on the right contains errors, correct the code enable SSB accomplish his task.   **Example activity 2** The Twinkledust fairy wants to cast a spell that will duplicate herself 5 times at different places on the screen. The following code has been provided, but it does not seem to work. Find the bug and correct it.	Debugging is an integral and important part of coding. While learners need to debug all the code the write, learners must also be provided with incorrect code which they need to debug. **Note:** It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively, using different activities and combinations of concepts and constructs. **Note:** Literature suggests that the biggest problem of novice programmers does not seem to be the understanding of basic coding concepts [in isolation] but rather learning to **apply** them [and combine them].to complete a task or solve a problem Therefore, at this level, beware of giving learners programming tasks that combine too many concepts (Robins, 2019).

Content (Grade 5 / Term 2)	Notes/Examples

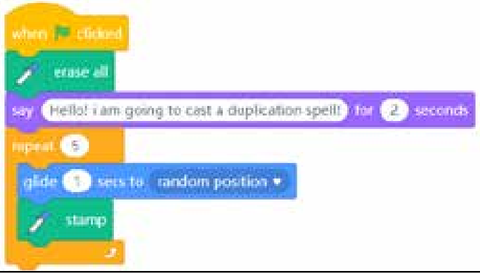
Possible Answer





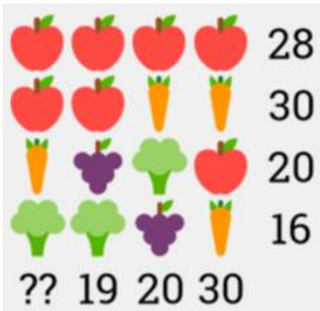
Jimmy your friend noticed that it seems as if after the code has been corrected and the Green flag is clicked again as the fairy keeps on duplicating. How should this be fixed?

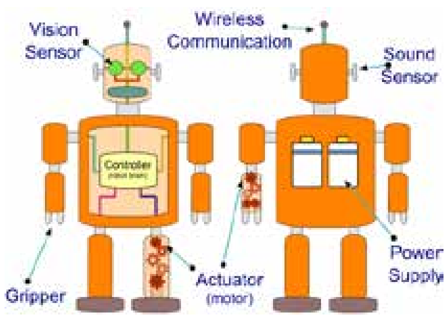
Possible Answer (second part)
Answer
Add an erase all block

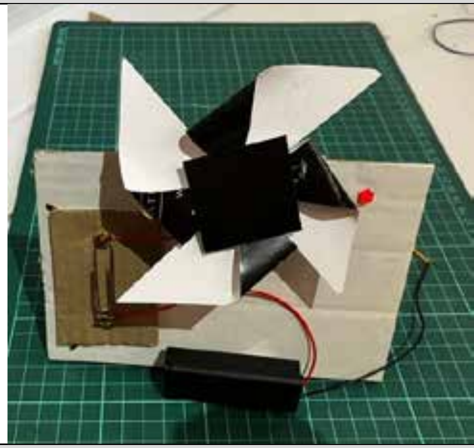


C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	Link to C.1, C.2 and D.6 and D.7L
Example activity In pairs, run the following code (on the right) in a block-based environment: Figure out what it does. Now, learners write similar code that uses keyboard input and changing sprite size, colour and other properties.	By identifying patterns, we can predict what will come next and what will happen again and again in the same way. In Computer Science/coding we analyse patterns in data and make predictions and generalisations based on the pattern analysis

Example activity 2
The numbers alongside each column and row in the drawing below are the sums of the values represented by the symbols within each column and row. Study the patterns and figure out what number should replace the question marks.



Content (Grade 5 / Term 2)	Notes/Examples
Robotics	
R.1 Explain what a robot is in simple terms.	Revise and extend from previous terms and grades R.1 – R.3 is done together At a basic level, learners need to acknowledge the following main components of a robot and what they are used for: • Sensors (for input) – a device that detects and responds to some type of input from the physical environment, e.g. light, motion, moisture, temperature, pressure, etc. • Controller (processor) Mechanical actions (actuator) (for output) – s mechanism that converts an electrical signal into a corresponding physical quantity such as movement. **Actuator** Mechanism that converts an electrical signal into a corresponding physical quantity such as movement, force, sound **Sensor** A device that detects and responds to some type of input from the physical environment, e.g. light, heat, motion, moisture, pressure
R.2 Identify different types of robots.	
R.3 Outline the different components of a robot	
 Pretty much like we humans receive inputs from our sensory organs, process them in our brain, and carry out the desired action; robots too have the same building blocks. The input to the robots is via sensors, the processing is done by the CPU unit, and then the desired output is obtained. Any robot is made up of three main parts – Sensors (for input), CPU (processor), and Mechanical Actions (for output). The sensory inputs that the robot takes can be anything from smell, touch, visual differences, etc. The central processing unit is the microprocessor or microcontroller that processes this input quantity, searches for the corresponding function to perform from the previously fed or programmed instruction set, and then sends the signal on to the output port. Upon reception of this signal, the robot will perform the desired action. Let us take an example of a robot that will stop once it meets any obstacle. **Robot: Obstacle Avoider** **Input**: Touch **Output**: Stop the motors **Purpose**: In this example, the robot is to move freely in any direction, and stop once it collides with any object. Once the robot meets any object, its ***input sensor (touch)*** will be activated. This sensor will send signal to the processing unit, as soon as it turns on. The CPU will look up in its instruction set to find the relevant action to be performed upon the reception of this signal.	
R.4 Present an understanding of how robots affect the world	**Link to R.1 – R.3**
Robots are fascinating machines that have a big impact on our world. Let's dive a little deeper into how they affect our lives. **Example activity** Provide learners with a KWLS chart / or they draw one in their workbooks. Learners write down what they already know and what they want to know about the topic Play a video on how robots affect the world, e.g. https://youtu.be/6_TpuJ3bnL8 and https://youtu.be/rBNzAGlSfnI Learners write down what they have learned and what they still want to know. Encourage learners to look for the information that they still want to learn. Discuss the aspects in the video and supplement with the information on the right. **Notes** • **Making Work Easier**: Robots are designed to perform tasks that are difficult, dangerous, or repetitive for humans. They can help assemble products in factories, perform surgeries in hospitals, or even assist in farming by planting and harvesting crops. This makes work more efficient and frees up human workers to focus on more creative or complex tasks. • **Saving Time**: Robots are super speedy! They can complete tasks much faster than humans. For example, in warehouses, robots can swiftly move heavy boxes from one place to another, making the process quicker and more efficient. Similarly, in our homes, robotic vacuum cleaners can zip around and clean the floors in no time, giving us more time to do other things we enjoy. • **Helping People**: Some robots are designed to assist people with disabilities or special needs. These robots can help individuals with limited mobility by fetching objects, turning on lights, or even providing companionship. Robotic prosthetic limbs can help people who have lost a hand or a leg to regain their mobility and independence.	Revise and extend from Grade 4 using different examples and activities. • **Making Work Easier**. • **Saving Time** • **Helping People** • **Exploring New**. • **Entertainment and Recreation**: **Note:** It is important to remember that robots are tools created by humans. They need humans to program them and give them instructions. While robots can be incredibly helpful, they are not meant to replace us. Instead, they work alongside us, making our lives easier, safer, and more enjoyable.

Content (Grade 5 / Term 2)	Notes/Examples
• **Exploring New** Frontiers: Robots have the amazing ability to go where humans can't easily reach. For example, rovers like NASA's Mars rovers explore the surface of Mars and send back valuable information about the planet. Underwater robots called ROVs (Remotely Operated Vehicles) can dive deep into the ocean to study marine life and explore underwater habitats that are too dangerous for humans. • **Entertainment and Recreation**: Robots are not only practical but also a lot of fun! You might have seen robot toys that can walk, talk, or even play soccer. Robots are also used in movies and shows, bringing fantastical characters to life. Additionally, robot competitions, such as robot sumo or robot races, provide exciting entertainment and encourage learning about robotics and engineering.	
R.5 Design a simple artefact based on a set of design instructions	**Link to R.6 and R.7**
Example activity - **Exploring Electricity, DC Motors, and Control Through a Fun Game** Design and create a simple electric circuit that powers a fan using a DC motor. The fan should be controllable with a switch, allowing it to be turned on and off. The design should focus on the functionality of the fan, its stability, and the effectiveness of the switch in controlling the fan's operation. 	Create a fun and interactive game using a DC motor, a light bulb, and a switch. This activity will help learners to delve into the basics of circuits, electricity, and robotics. Instead of coding, we will use a hands-on activity to engage students and explore the principles of robotics. **Note**: Evidence suggests that pupils should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practice (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).
R.6 Mimic the operations of a robot	**Link to R.5 and R.7**
Example activity 1 The fan built in R.5 moves, mimicking the operations of a robot. **Example activity 2** Design a grid activity (in block-based coding environment) with a virtual robot and obstacles. Virtual robot then follows instructions to avoid obstacles and/or react in specific ways when it touches the obstacles) (Can also be done physically on the floor where learners act as robots)	Use virtual robot in block-based coding environment
R.7 Create, test and execute a set of robotic instructions	**Link to R.5 and R.6**
Example activity Using computational thinking, design thinking and the engineering design process: • Plan, create a basic fan as described in R.5. • Execute and test the design and debug/fix if required.	Do with R.5 Learners need to use computational thinking and design thinking while following the steps on the left when they design an artefact based on a set of design instructions (R.5)
Digital Concepts	
D.2 Recognise that he or she is living as citizens in a digital world.	

Content (Grade 5 / Term 2)	Notes/Examples
Reinforce and extend from the previous grades **Example activity: Cyberbullying** Introduce cyberbullying asking learners if they have heard about it and what it is. Discuss the seriousness of cyberbullying and how it can lead to emotional distress, anxiety, and even harm to the victim's mental health. Provide examples of cyberbullying and engage learners in a discussion about the consequences of cyberbullying on the victim, the bully, and the broader community. Now, divide the learners into small groups. Provide each group with a different cyberbullying scenario handout. Instruct the groups to discuss the scenario and come up with an appropriate response and coping strategy for the victim. After the discussion, have each group present their scenario and share the coping strategy they developed.  Emphasize that cyberbullying is never acceptable, and everyone has a responsibility to create a safe and respectful online environment. Ask learners to share their thoughts on how cyberbullying might make someone feel and how it can impact their daily life and well-being. Reinforce the importance of empathy, respect, and responsible behaviour online Encourage learners to be upstanders, not bystanders, and to support anyone facing cyberbullying. Remind learners that they can always seek help from adults if they or someone they know experiences cyberbullying Learners now write a short reflection on the lesson, sharing their thoughts on cyberbullying and how they can contribute to a safer digital environment for themselves and others.	Cyberbullying is use of technology, such as social media, instant messaging, or email, to harass, intimidate, or humiliate someone repeatedly. Learners need to understand that cyberbullying includes • Sending mean messages or comments online • Spreading rumours or lies about someone through social media • Creating fake profiles to mock or embarrass someone • Sharing embarrassing photos or videos without permission • Coping strategies for deal with cyberbullying • The consequences of cyberbullying Present coping strategies for dealing with cyberbullying. Some strategies include: • Do not respond or retaliate to cyberbullying messages; it can escalate the situation. • Block and report the cyberbully on the platform where the bullying is occurring. • Save evidence of cyberbullying, such as screenshots or messages, for reporting purposes. • Reach out for help and support from a trusted adult, parent, teacher, or school counsellor. • Encourage open communication and empathy among peers to build a supportive school community. • Promote positive online behaviour and remind learners about the impact their words can have on others.
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.4 and D.5**
Link to computing devices as part of an ICT system	Done with D.4 and D.5
D.4 Identify the common uses of ICT in the real world	**Link to D.5**
Briefly remind learners that IT mostly deals with computing and data and information management, whilst ICT adds 'communication' – being able to exchange data and information over networks. ICTs are used to communicate with people all over the world. Identify everyday uses of ICTs, e.g., mobile phones (communication). Provide examples of how ICT improves communication, e.g., WhatsApp, and social media. In schools, teachers use ICT to teach learners by showing them videos or pictures on a computer or whiteboard or digital projector. Ask learners to provide more examples of ICTs in their daily lives and discuss these examples and the difference between ICT and IT. Basic understanding of a network (e.g., school network / entertainment / shopping as an example) **Example activity: Basic concept of networks** Using an example such as mobile phones that communicate with each other via cell phone towers (wireless connection), computers can also communicate via a network (physically connected or wireless). Draw a simple diagram of a network on the board, showing several devices connected to each other. Emphasize that networks allow devices to share information, files, and resources, making it easier for people to communicate and work together. Where do we find networks? At Home: Explain that this type of network is found in homes and connects devices within a family, allowing them to share internet access and files. At School: Mention that schools have networks connecting computers, printers, and other devices, making it easier for learners and teachers to share resources.	Reinforce and extend from previous Grades and terms. using different examples and activities. Learners should be able, at a basic level to explain what a network is and understand that computing devices in a network can communicate / send data and information over the network. Use examples that learners will understand, e.g., school network or a cellphone network.

Content (Grade 5 / Term 2)	Notes/Examples
Use an analogy of sending letters by passing notes from one learner to the other until the note gets to the intended recipient to explain how data contain information and are sent from one device to another until they reach their destination. Emphasize that devices in a network need unique addresses to identify them, like how people have unique addresses for their homes.	
D.5 Differentiate between the components of an ICT system	**Link to D.3, D.4 and D.7**
Reinforce and extend from the previous term (Refer to Grade 4 Term 2 D.5.) Discussion about the differences between components and their functions. Learners need to understand that the basic components of an ICT system. • Hardware (e.g., computers) • Software (e.g., applications/ programs) • Data (e.g., files,) • Networks (e.g., internet) • People (e.g., users,) **Example activity: Computer components carnival – exploring components.** Introduce the concept of computer components and explain that today, they will embark on an exciting journey through the "Computer Carnival" to learn about components. Divide the classroom into several stations, each representing a different computer component. Allow learners to create cards for their components. Include the description of the component, what the function is and how it works. Allow groups to present their components. End the lesson with a fun quiz game related to computer components. Create multiple-choice or true/false questions based on the information presented during the "Computer Carnival." For an extended creative project, learners can work in pairs or small groups to design and create their own "Computer Carnival" posters or exhibits. Each group can focus on one computer component and create an attractive and informative display to showcase.	Learners should recognize the components of technology, such as hardware (e.g., computers, smartphones, tablets) and software (e.g., applications, operating systems). They should understand how these components work together to deliver technological solutions. An ICT system is made up of computing devices (e.g., computers), programs (the instructions that tells the devices what to do), data and information and networks (including the internet) that allows the devices to communicate and send data and information as well as the people that use all of this.
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.2**
Ask learners what they think technology is and write down their answers on the whiteboard or flip chart paper. After collecting their responses, provide a simple definition of technology, such as "Technology refers to tools, machines, and techniques used to solve problems and improve our lives." **Example activity: Technology in the world we live in** Divide the learners into small groups of 3-4 learners. Assign each group a specific area of technology, such as communication, transportation, or entertainment. Ask each group to brainstorm and discuss examples of technology in their assigned area. They should also think about how those technologies have impacted society and improved our lives. Ask each group to present their findings to the rest of the class. As they present, facilitate a brief discussion on the impact of each technology and how it has changed the way we live. Indicate that technology has both positive and negative effects. Explain to the learners that information technology (IT) is a branch of technology that focuses on the use of computers and software to store, retrieve, transmit, and manipulate data or information. Discuss how IT plays a crucial role in various aspects of our lives, such as communication, education, entertainment, and business. Depending on the available resources, you can ask learners to work individually or in pairs on the following: Research and create a one-page report on a specific technology (e.g., smartphones, social media, online learning platforms) and its impact on society. Gather the learners back together as a whole class. Ask them to share their findings from the application exercise or display their reports. Facilitate a discussion on the importance of responsible use of technology and its potential benefits and drawbacks.	
D.7 Present a basic understanding of the concept of input processing and output.	**Link to C.2 – C.5**
Revise Input-Processing-Output from previous grades and terms. **Example activity: IPO table using coding** Learners develop an algorithm for a daily activity, indicating, the input, process and output.	Do with C.2 – C.5 IPO can be done when learners develop algorithms/programs in the block-based coding environment (C.2). IPO can be done when working on devices, emphasising input and output devices (processing between input and output) and storage for later use (store their files in their folders for later use)
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to Cs and Ds**
Reinforce and extend from the previous grades and terms using different examples and activities. Align to activity in D.3, D.7.	Done in relation to & Link to C.1 – 7

3.2.3 Term 3

Content (Grade 5 / Term 3)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to all Cs**

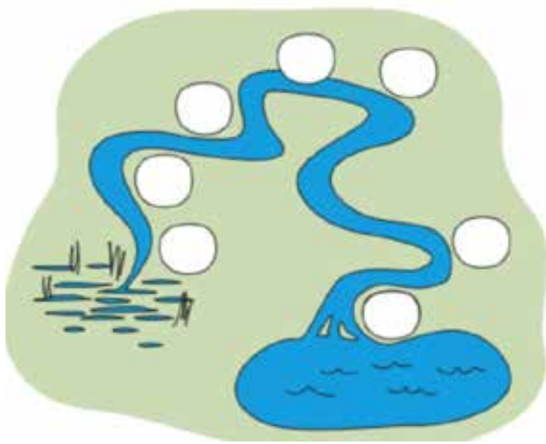
Example activity Use CT

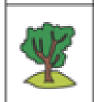
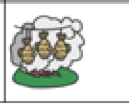
Read through the description below, study the picture on the right and then use computational thinking to answer the questions that follow

A river flows from a swamp to a lake. Halfway down the river is a pier with a cafe. Visitors at the cafe enjoy a view of branches on the river next to the pier. At the cafe visitors can also book a steamboat trip to the carousel which is located at the mouth of the river. On the return trip from the lake to the pier the steamboat will stop at a barbecue near the carousel for a lunch of smoked fish. As the steamboat returns to the pier, to the left, visitors will enjoy a view of the hundred-year-old oak tree which is in the swamp and a hill which is located between the pier and a windmill.

Question:

Place the correct letters for each landmark in the correct order to indicate their location along the riverbank, starting at the swamp and ending at the lake.



A. oak tree	B. barbecue	C. windmill	D. carousel	E. branches	F. pier	G. hill
						

Write the letters of the correct answers in the empty circles in the picture above. 2018-TS-JUNIOR-Q-paper.pdf (olympiad.org.za)

Notes/Examples:

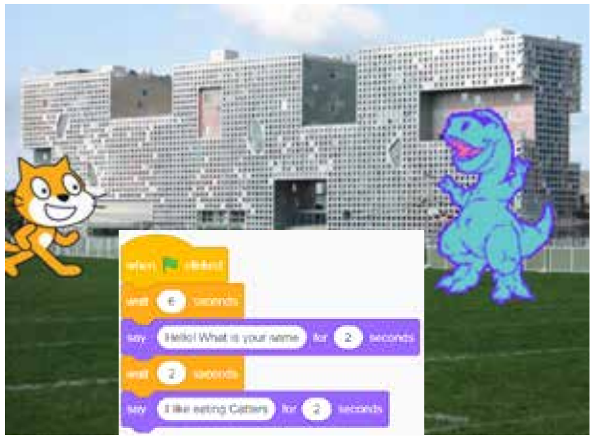
Abstraction helps to focus on the important information in the description of the problem as it helps to reduce complexity.
Decomposition helps to break the problem in smaller sub-problems that makes it easier to understand and approach the problem by solving each induvial part.
Pattern recognition helps to find patterns and relationships among parts which helps in solving the problem.
Algorithmic thinking helps us to create a clear and logical pathway to reach a solution

Note:
Often, in real life or as a programmer, we get information from another people's chaotic description. Then we must change a vaguely described sequence of actions into an exact and logically ordered sequence to complete a task or develop a computer program. People often describe their ideas of what the program should do by lengthy sentences and descriptions and often do not use a clear order in which things must happen. Therefore, a programmer must be ready to transform such descriptions into a more exact form.

Content	Notes/Examples
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	**Link to C.1, C.3 – C.7 and R.6 – R.7**
C.3 Interpret and execute a given symbolic or written set of commands	**C.2 and C.3 done together**

Example activity 1 – Using 2 sprites that interact

Provide learners with the code for both sprites and let them first inspect the code for both sprites and explain what it does.
They then run the program and compare what is happening with their interpretation.
You need to click the green flag at the top to execute both sprites' code.



Notes/Examples:

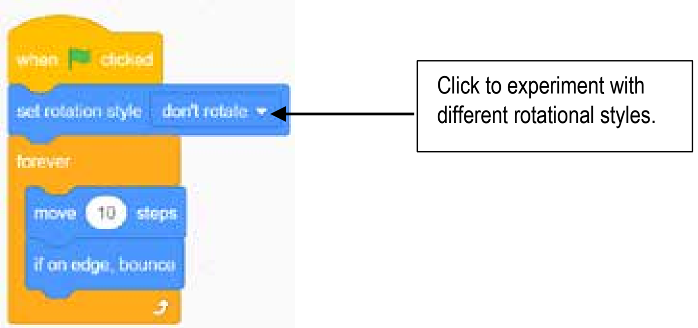
Note:
Interaction between two sprites is made possible using the time laps between the 2 sprites

Note:
It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively, using different activities and combinations of concepts.

Note:
Provide learner with activities enabling them to
- read code and explain what it does or
- work through (trace) / act out code (physically or simulated /pen-and-paper) to determine the output or the correctness or

Content (Grade 5 / Term 3)	Notes/Examples

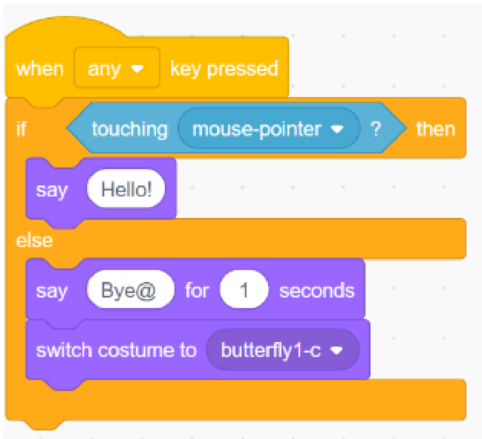
Example activity 2 – Introduce Set rotational style
Provide learners with the code below
Let them run the code and explain what it does
Let them click on the drop-down arrow and experiment with the different



Example activity 3 – If touching mouse pointer
Provide learners with the code on the right.
Learners study and execute the code, then explain what it does,

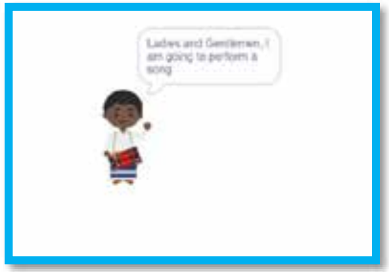
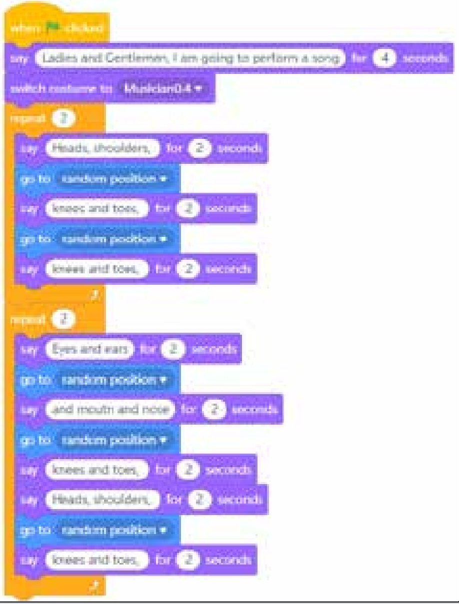
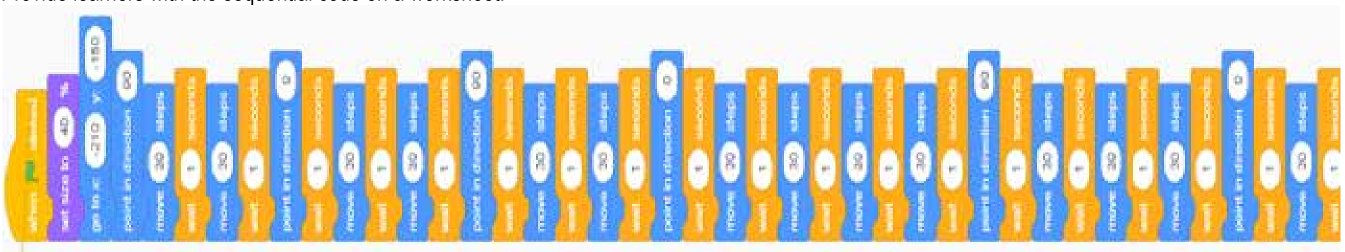
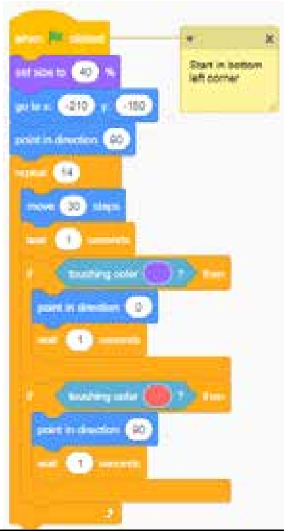
Learners now code the following algorithm:
When the user presses any key, then if the mouse pointer touches the sprite, it should greet the user, else it should say "Bye" to the user and switch costume

Possible solution for activity 3 (algorithm – touch mouse pointer)



Notes/Examples

- provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or
- translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions))
- add some functionality/instructions to an existing program.
- rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or
- choose the correct solution from 2-3 options or
- compare different solutions to evaluate efficiency or
- debug an algorithm or block-based program (find the bug, describe the bug and correct it)
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

Content (Grade 5 / Term 3)	Notes/Examples
Example activity 4 – open-ended Learners use their knowledge, skills and experience to design, code, implement, test and debug a program of their choice	
C.4 Debug a given symbolic or written set of instructions	**Link to C,1 – C.3 andC.5 – C.7 as well as R.5 – R.7**
Example activity Fatima wants to sing a song for the class. The code written for the popular song: *Head's Shoulders, Knees, and Toes*. When the code is executed, the song is not sung correctly. How should the code be changed to have Fatima sing the song correctly? 	**Note:** Learners generally struggle to debug. Teach them to persist. **Note:** Concrete activities remain important as literature suggests that the primary weakness of today's pedagogy of programming is that it doesn't provide enough opportunity for the novice to develop concrete operational skills, via the correct types of exercises…due to too much emphasis on writing large amounts of code, and problem solving.
C.5 Evaluate a given solution towards potential improvement	
Example activity 1 – Improve code Learners wrote sequential instructions (code) in Grade 4 (bug walk). Learners work in pairs. Provide learners with the sequential code on a worksheet:  Learners need to improve the code by identifying patterns / code that are repeated and could be placed in a loop to reduce the number of instructions using pen-and-paper. Inform learners that they could use the following to improve the above solution: Repeat (14x) and If touching colour. After planning the improved solution, learners code the improved solution, execute the code, test, and debug until it works.	Learners are provided with code which they need to improve. Refer to the activity in Term1 R.6 **Possible solution** Learners need to compare the solutions and explain why the solution using a loop is a better solution. It is very important that learners understand how to look for patterns to improve code / write better solutions 

Content (Grade 5 / Term 3)	Notes/Examples
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations	**Link to C.1 = C.5 and C.7 as well as R.5 – R.6**

Example activity 1

Provide learners with the following code.

Let them work through the code, line-by-line, explaining each line of code.
Now, let them explain wat the program does.
Learners now run the code and compare with their interpretation.
Now, add another sprite (elephant)
Copy the code to the new sprite (elephant)
For the new sprite, let them change the code so that the sprite walks a rectangle.
(only 2 changes need to be made)

Notes/Examples: Learners study code to see what it does and then write a similar program for another situation

Learners change existing code to enable another effect.

C.7 Create or complete a pattern to represent a data set	**Link to C.1**

A Latin square has two important properties:

- A row or column never contains the same figure/number twice
- Every row and column contain the same figures/numbers

On the right-hand side is an example of a full Latin square

1	2	3	4
2	3	4	1
3	4	1	2
4	1	2	3

For each of the Latin squares below (A-F), figure out which of the four numbers belongs in the place of the question mark.

1	2	3	4

A

	1	3	
1		2	
	2	4	
4		?	2

B

			2
			1
		4	?

C

		1	
		4	
	1	3	
	?		3

E

3			2
4			3
			4
?			

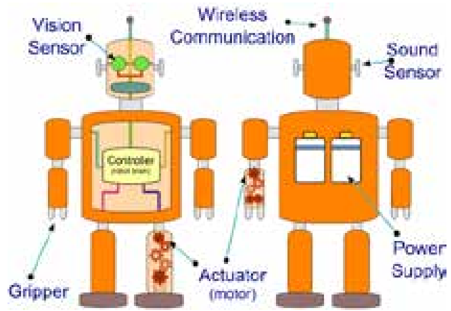
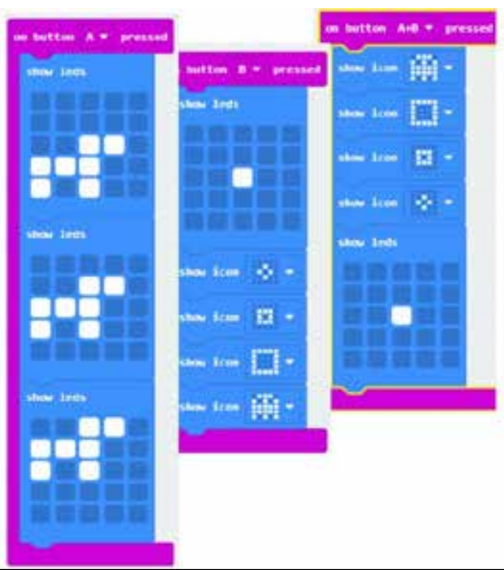
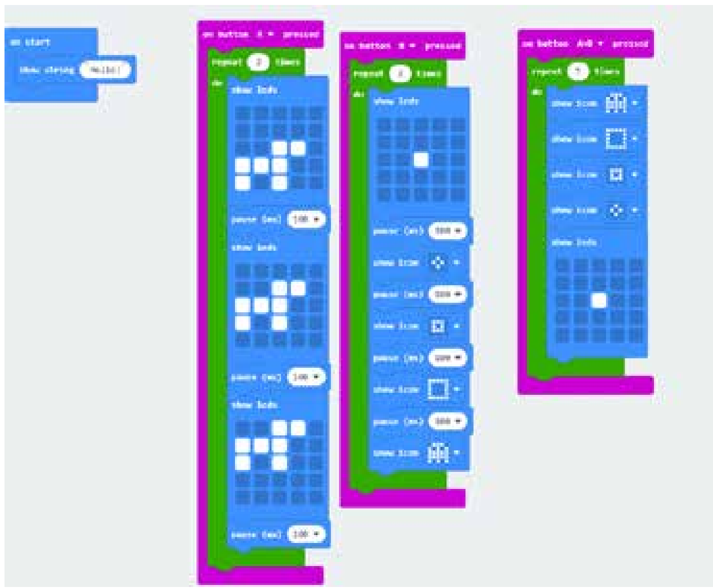
E

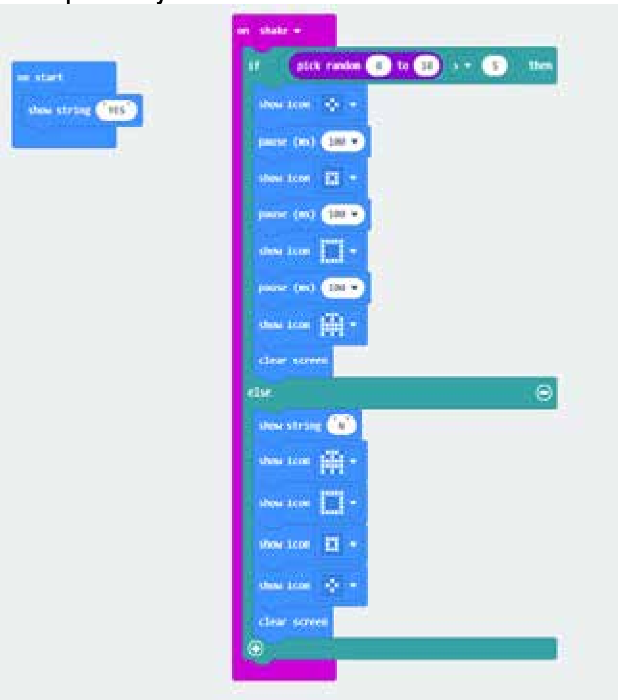
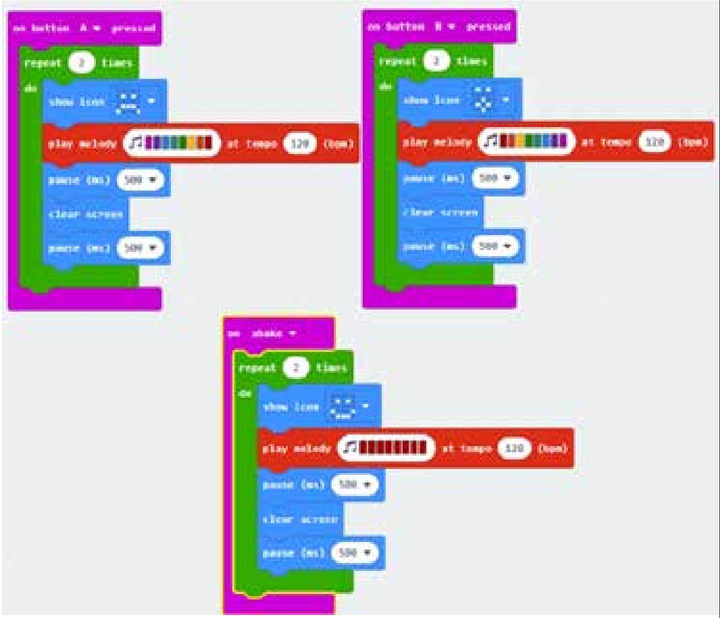
2		4	3
	2		4
		2	?

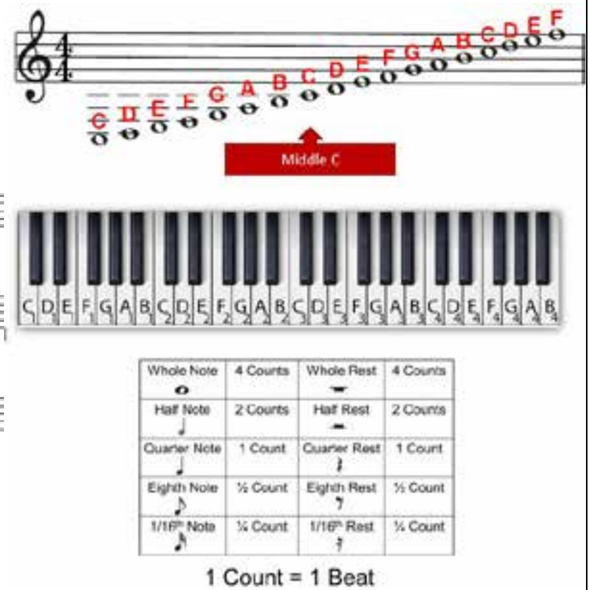
F

	?		
2		3	
	2	4	
		2	3

Robotics	**Link to R.1 – R.6**
R.1 Explain what a robot is in simple terms.	R.1-R.3 is done together
R.2 Identify different types of robots.	
R.3 Outline the different components of a robot	
Briefly revise aspects of R.1 and R.2 Discus and explain the following, linking to R.5, R.6 and R.7:	

Content (Grade 5 / Term 3)	Notes/Examples
Sensors: These are like the robot's senses. Sensors help the robot gather information about its surroundings. They can detect things like light, sound, temperature, and distance. For example, a robot might use a sensor to "see" if there's an obstacle in front of it or to "hear" a sound. **Communication**: Just like we talk to each other using words, robots need a way to communicate too. They use special devices to send and receive information. It's like how we use a phone to call someone or send a message. Robots can communicate with their human operators or with other robots to share important information. **Grippers and Attachments**: These are like the robot's hands or tools. Grippers are used to grab and hold objects, just like we use our hands to pick up things. Robots can have different types of grippers depending on what they need to do. Some robots might have a claw-like gripper, while others might have a suction cup or even a magnetic attachment.  **Actuators**: Actuators are like the muscles of a robot. They help the robot move or perform certain actions. For example, a motor is a type of actuator that can make a robot's wheels turn or make a robot's arm move up and down. Actuators help the robot do the tasks it's programmed to do. **Controllers**: Controllers are like the brains of a robot. They tell the robot what to do based on the information it receives from sensors and commands from its human operator. Controllers are like the robot's instructions or rules that guide its actions. They make sure the robot knows how to respond to different situations.	
R.5 Design a simple artefact based on a set of design instructions	**Link to R.5 and R.7 and D.10**
Project **R.6 Mimic the operations of a robot** **Example activity 1** Introduce the coding environment taking learners through the virtual environment or using a video, e.g. https://youtu.be/46nqNwqVE1w **Example activity 2** Write code to display the following images using different inputs (Button A pressed, Button B pressed and Button A+B pressed):  **Example activity 3 Input, sequence and basic loop** 	Start with D.10 to introduce the environment. Introduce the micro controller programming environment. **Note:** Learners need to transfer the programming knowledge and skills acquired in the block-based coding environment (use their experience with another environment) to the new micro controller block-based coding environment. It is a good idea to quickly revise the knowledge and skills required for the new coding environment by linking it to the first coding environment learned and explain how it works in the new environment. **Note**: In Grade 5, learners do not work with the physical microcontroller, but only use the virtual one on the screen. **Note**: Evidence suggests that pupils should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practice (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).

Content (Grade 5 / Term 3)	Notes/Examples
Example activity 4  **Example activity 5** 	
R.7 Create, test and execute a set of robotic instructions	**Link to R.5 and R.6**
Basic outline of how a robot is coded to perform tasks. • **Define the Task**: The first step in coding a robot is to decide what task or action you want the robot to perform. For example, you might want the robot to move forward, turn, or pick up an object with its gripper. • **Plan the Steps**: Once you know the task, you need to plan the steps or actions the robot should take to accomplish it. Break down the task into smaller actions. For example, if the task is to move forward, the steps might be to activate the motors in the robot's wheels. • **Write the Code**: Coding is like giving instructions to the robot in a language it can understand. You write the code using a programming language, which is a set of commands and rules that the robot can follow. Each command tells the robot what action to take. For example, you might write a line of code that tells the robot to turn on the motors and move the wheels forward. • **Test and Debug**: After writing the code, it's time to test it and see if the robot performs the desired task correctly. Sometimes there may be errors or bugs in the code that cause the robot to behave differently than expected. This is called debugging. If something doesn't work as intended, you can go back to the code, identify the problem, and make corrections. • **Upload the Code**: Once the code is working correctly, you need to upload it to the robot's controller. This is usually done using a computer or a special device that connects to the robot. The controller will receive the code and start executing the instructions. • **Execute the Task**: Once the code is uploaded, the robot's controller will interpret the instructions and send signals to the actuators, such as motors or grippers, to perform the desired actions. The robot will start executing the task based on the code you wrote. • **Refine and Iterate**: Sometimes, the robot may not perform the task perfectly on the first try. That's okay! You can refine and iterate on the code to improve the robot's performance. You can adjust, add new instructions, or change the sequence of actions to make the robot perform the task better.	Learners write code in a block-based coding environment using a micro controller to play a song.

Content (Grade 5 / Term 3)	Notes/Examples
Example activity Use the steps as outlined above and the information provided, then write the code to produce music for the song, Bana ba Sekolo, The music for the song must play when Button A is pressed.   **This activity allows learners to also interpret symbols and recognise patterns**	
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.1, D.3, D.4, D.5 and D.7,**
Reinforce and extend from the previous grades and terms using different examples and activities. Using their knowledge skills and experience gained, learners create concept maps to illustrate and summarise the following: Technology → Information technology → Information and Communication Technology (ICT) Computing devices (including the concept of input, processing and output) **Quiz** As a concluding activity, provide learners with a quiz (using apps such as Kahoot!, Google forms, MS Forms, Quizlet, etc.) about what they have learned about technology, the purpose of IT.	D.1, D.3, D.4, D.5 and D.7 are done together by reinforcing concepts learned from previous Grades and terms

Content (Grade 5 / Term 3)	Notes/Examples
D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.6**

Reinforce and extend from the previous grades and terms using different examples and activities.)
Review caring applicable to the device in use, e.g., tablet, computer, microcontroller.

Example activity 1: **Combining choice structures with digital citizenship.**
The teacher introduces life choices, e.g. Will I greet my friend? Will I be friendly today? Will I obey the rules?
Using think-pair-share, (two learners sitting next to each other) to come up with one rule for good citizenship that they can remember.
The teacher gives an example of a drawing showing a road that splits into a fork and explain that when one reaches a fork in the road, one needs to decide which way to go e.g., doing homework? or Watching TV? (Just like the IF…THEN construct when doing coding)

Note: for this activity the one choice must include a positive outcome and the orher a negative outcome

The small group should now draw their own pathway for a good digital citizenship rule and write positive outcomes and negative outcomes. One learner writes the positive outcome, and the other learner writes the negative outcome

One can extend the activity by letting the road split into another fork, with another choice and a positive outcome and a negative outcome.
Allow learners to share their choices and the possible outcomes they came up with.
The teacher revises the concepts and adds to the discussion and addresses misconceptions.

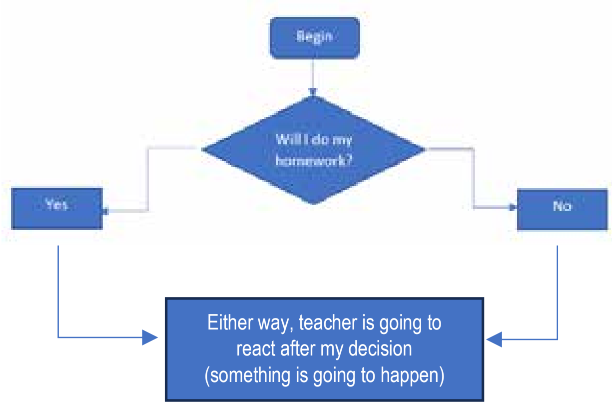
Example activity 2: Different outcomes game
Play a game by starting to pose critical decisions to players. The main objective is for players to explore the different story paths by making critical decisions that define their journey through a made-up world. Each path offers unique challenges, experiences, and outcomes, allowing players to replay the game to discover all the possible storylines.
See game Common Sense:: Digital Compass :: Game.

Notes/Examples:

Reinforce and extend from previous Grades and terms using different examples and activities.

We must make choices every day, some with positive consequences and others with negative consequences

Link to C.2 – C.5: IF…THEN….ELSE coding structure



Content	Notes/Examples
D.8 Interpret a pattern to represent or communicate a message or image	**Link to C.1, C.6 and C.7**
D.9 Create a pattern to represent or communicate a message or image	**D.8 and D.9 can be done together**

Example activity
Provide learners with a square containing the letters of the alphabet.
Explain to them that this cipher uses lowercase letters and number to encrypt and decrypt messages, e.g., the world HELLO is encrypted as b3a5c2c2c5
Note: there can be variations of the grid, e.g., a different way of ‘packing’ the alphabet or both rows and columns could be numbers or both rows and columns could be letters.
Divide learners into pair.
Each pair encrypt a message using a grid.
Pairs swop their cipher grids with the and the encrypted message and decrypt the message

	1	2	3	4	5
a	A	B	C	D	E
b	F	G	H	I	J
c	K	L	M	N	O
d	P	Q	R	S	T
e	U	V	W	X	Y/Z

Notes/Examples:

Learners need to know
- Encryption - a process of encoding messages to keep them secret, so only "authorized" parties can read it.
- Decryption - a process that reverses encryption, taking a secret message and reproducing the original plain text.
- Cipher - the generic term for a technique (or algorithm) that performs encryption.

Content	Notes/Examples
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to R.5**
When working on the computer in the block-based coding environment, guide learners to become familiar with the environment (Link to R.5 – R.7) Teacher explicitly guides learners through the process of switching on, opening the new block-based coding application and to understand that they work in a new integrated development environment (IDE) Open the microcontroller app	Introduce the block-based robotics application environment. This must be done in relation to R.5 **Note:**

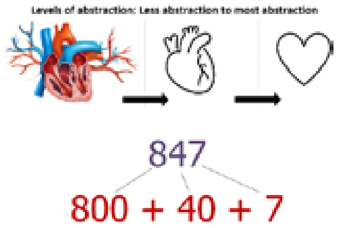
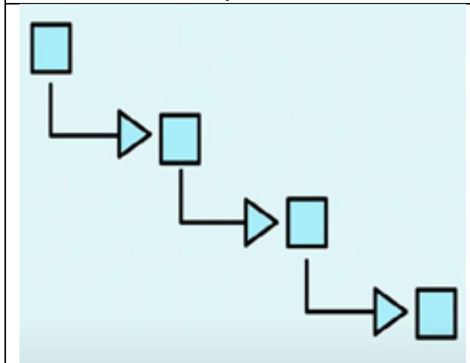
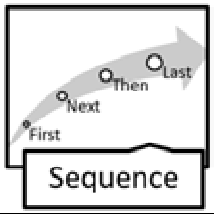
Content (Grade 5 / Term 3)	Notes/Examples
Open a new project and give it a name. Now study the IDE. You will see the • Microcontroller simulator with LED matrix (simulates/outputs the result of your coding on the microcontroller). • Toolbox (where you find your coding blocks) • Workspace (where the coding happens) You the Welcome pop-up to guide you through the above main areas of the app  Show and explain on a just-in-time basis – what they will need at a particular stage or for completing a specific activity. Learners can now start developing their first microcontroller app (Refer to R.5) **Note: Learners only work with the microcontroller simulator on the screen. The physical microcontroller is only introduced in Grade 6**	Do when the robotics coding environment is introduced (move to term 1 if coding (C.2 – C.5) and robotics (R.5 - R.7) is done in parallel) Teacher guides the learners • the main parts of the IDE • the new blocks that will be used • how to navigate the new environment • tutorials for the new IDE • Reinforce and extend file extensions and file management to align to the new file type used by the IDE. • Basic file management (open and save program files) Link concepts to the block-based coding IDE that learners already know and explain to them that the coding concepts that will be used here are like what they have learned in coding.

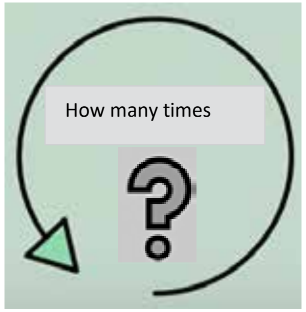
3.2.4 Term 4

<table>
<tr><th>Content (Grade 5 / Term 4)</th><th>Notes/Examples</th></tr>
<tr><td>Coding</td><td></td></tr>
<tr><td>C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.</td><td>Link to C.2 – C.7 and R.5 – R.7</td></tr>
<tr><td>Provide learners with a quiz (in Google/MS Forms/Kahoot!/ etc.) to test their understanding of computational thinking.
Example activity
For each of the following, indicate if it refers to Abstraction, Decomposition, Pattern Recognition, Algorithm (or maybe more than one):

Your timetable
Cleaning your room by packing away your clothes, then making your bed, then dusting and then vacuuming the floor.
Baking a cake following a recipe.
A plan of the school grounds
Noticing that all birds have feathers, two wings, a beak and two legs.
Directing someone from your home to the nearest shopping centre
You need to fetch 10 l water from the river to your house in the village. You know that you are not strong enough to carry one container with 10l water. You decide to use a 5l container and doing two trips.
Realising that the difference between terms in a series of even numbers is two, e.g. 10, 12, 14, 16 …
</td><td>People use computational thinking all the time

One can also include pictures as part of the quiz:



</td></tr>
<tr><td>C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.</td><td>Link to C.3</td></tr>
<tr><td>C.3 Interpret and execute a given symbolic or written set of commands</td><td>C.2 and C.3 are done together</td></tr>
<tr><td>By now, learners need to understand the following:
<table>
<tr><th>Sequence</th><th>Selection</th><th>Repetition</th></tr>
<tr><td>
A series of actions are performed in a specific order, the first action first, then the second action, then the next action until the last action.

</td><td>
There is sometimes more than one path to follow, and we need to ask a question to decide which path to follow

</td><td>

Loops also ask questions, such as how many times? but ask the question over-and-over again and perform actions over-and-over until the condition is satisfied

</td></tr>
</table></td><td>Note:
Note
Sequence –Learners must now be able to write basic code in the correct logical sequence.

Learners must now be able to use the following control structures:

Selection – learners must now be able to do a basic
IF…THEN
IF….THEN….ELSE
Based on a simple condition

Repetition – learners must be able to use a

Forever loop
Repat loop with a fixed number (constant) of iterations

It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively, using different activities and combinations of concepts.

Also refer to Table 2-7 for other coding concepts that learners must be able to use.</td></tr>
</table>

<table>
<tr><th>Content (Grade 5 / Term 4)</th><th>Notes/Examples</th></tr>
<tr><td>

Example activity 1 – forever spike pattern
In pairs, provide learners with the code on the right.
Leaners run the code and observe what it does.
Learners then study the code and discuss its working.
Learners explain to each other to see if they understand how it works.
Learners then use this example to write their own code to achieve a similar outcome.

Example activity 2
Provie learners with a programming task where they must use some of the coding they have learned so far (combining 2 or three concepts, however, keep it manageable)

Example activity 3 – open-ended
Learners use their knowledge, skills and experience to design, code, implement, test and debug a program of their choice.

</td><td>

Note: Provide learner with activities enabling them to

- read code and explain what it does or
- work through (trace) / act out code (physically or simulated) to determine the output or the correctness or
- provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or
- translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions))
- add some functionality/instructions to an existing program.
- rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or
- choose the correct solution from 2-3 options or
- compare different solutions to evaluate efficiency or
- debug an algorithm or block-based program (find the bug, describe the bug and correct it)
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.
- develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.

</td></tr>
<tr><th>C.4 Debug a given symbolic or written set of instructions</th><th></th></tr>
<tr><td>

Example activity 1
What is wrong with the following code? Explain what is wrong, then correct the code.

</td><td></td></tr>
<tr><th>C.5 Evaluate a given solution towards potential improvement</th><th></th></tr>
<tr><td>

Example activity – Improve code
The code on the right uses two IF statements.
Change the code to use an IF…ELSE statement instead, but it must still provide the same output/ solve the same problem.
The program must still have the same outcome.

</td><td>

Improve code below using IF…THEN…ELSE (show grade 4 code and give new code – compare and explain the difference

</td></tr>
</table>

Content (Grade 5 / Term 4)	Notes/Examples
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to C.1 – C.5 and C.6 and R.6**

Example activity

ASCII is a character encoding standard for electronic communication. ASCII codes represent text in computers.

A computer sorts characters and strings (words) according to ASCII values.

The ASCII-value of letters are as follows:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122

ASCII values of special characters:

- The ASCII value for the **space** is **32**
- ASCII value of the **apostrophe** (') is **39**
- ASCII value of hyphen (**-**) is **45**

Arrange the following surnames alphabetically according to the ASCII-values:

Nel, McCracken, Dada, de Lange, le Clerque, Mudau, van Buuren, De Lange, Vandeventer, Delport, MCDonald, Nell, Van Deventer, O'Niel, Naidoo, Mini, StBernard, Leclerque, Nel-Pieters, Olwage

Notes/Examples:

Learners use decomposition.
First order the surnames alphabetically (normal alphabet)
Then separate the surnames starting with uppercase letters from those with lowercase
Now consider special characters

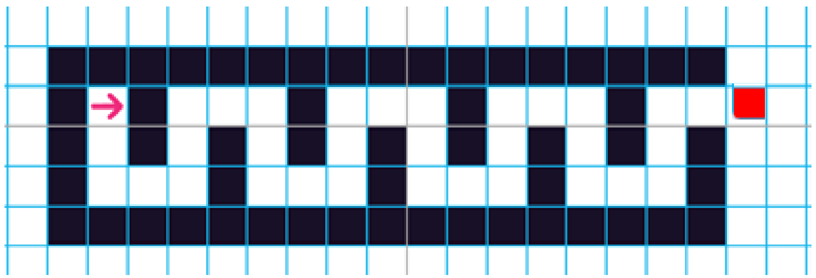
Content (Grade 5 / Term 4)	Notes/Examples
C.7 Create or complete a pattern to represent a data set	**Link to C.1 – C.6 and R.6**

Example activity 1

Help the robot to exit the maze. The robot can only act on the following commands:

F = move one block forward (in the direction it is facing)
R = turn right (stay in same block, face in walking direction)
L = Turn right (stay in same block, face in walking direction))

Adapted from: [2017-TS-JUNIOR-Eng-Q-paper.pdf (olympiad.org.za)]



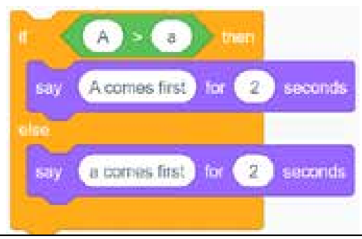
Write a set of instructions to help the robot exit the maze using the above instructions. The robot starts on the left top block, facing in the direction the arrow is showing and must end on the red block, facing in the same direction as it started.

Identify the pattern for moving through the maze.

How many times does the pattern repeat?

Activity 2 – Which is bigger?

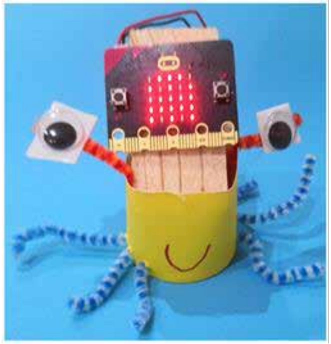
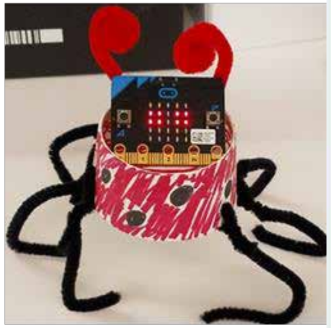
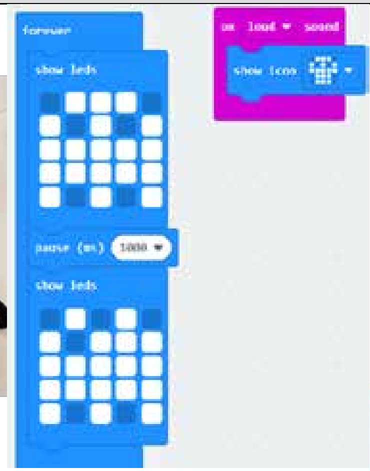
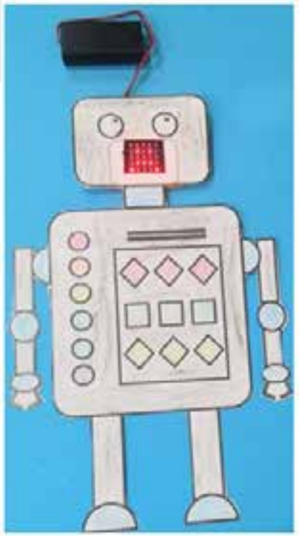
Study the code on the right. What would the output be?
Explain your answer.
Now, run the code and see what the output is.
Explain the output.

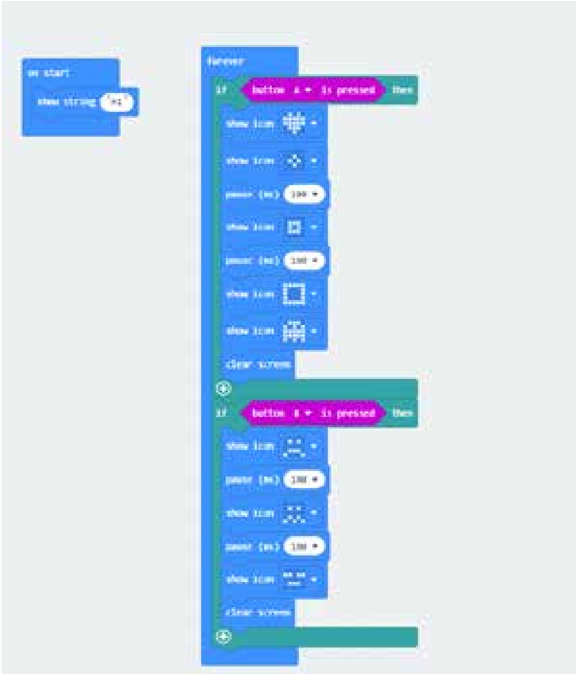
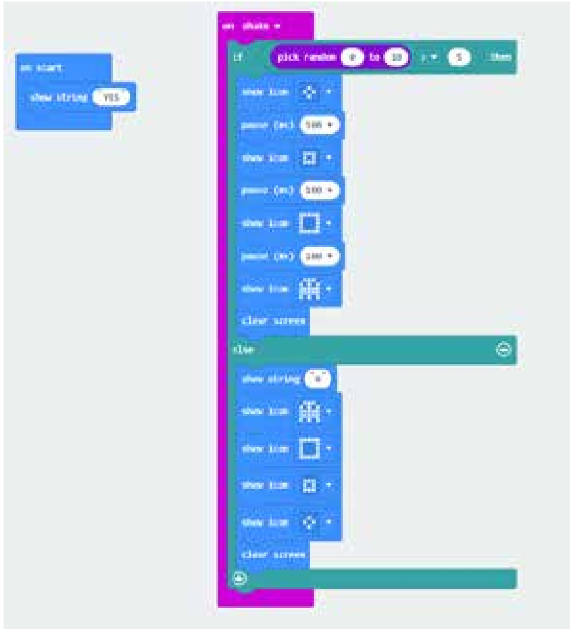
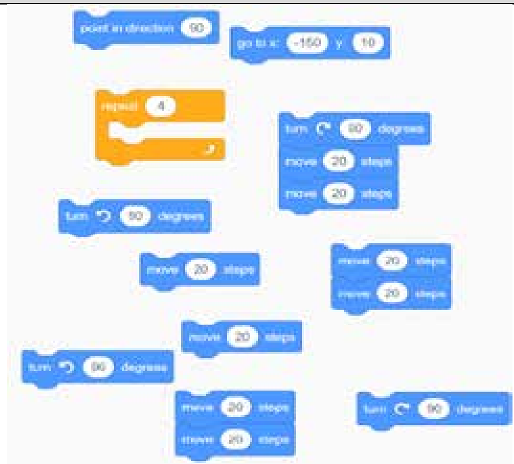


Notes/Examples:

This activity is implemented in a block-based coding environment to mimic the operations of a robot (R.6).
However, it is important that learners first do it as a pen-and-paper activity in C.7 as it then serves as the planning part for the coding activity.

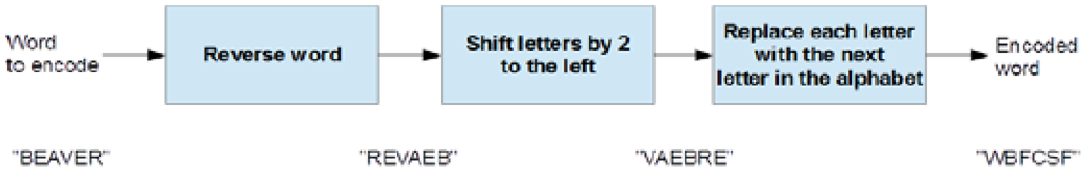
Learners need to explain the output based on what they have learned in C.6 (ASCII codes)

Content (Grade 5 / Term 4)	Notes/Examples
Robotics	
R.4 Present an understanding of how robots affect the world	**Link to R.1, R.2 and R.3**
Example activity – Robots and its relationship to AI + Basic definition AI, which stands for Artificial Intelligence, is when machines or computers are designed to think and learn like humans. It's like giving them a brain to make decisions and solve problems on their own. AI is used in many different things, like computer games, smartphones, and even robots. **Relationship between AI and robotics.** Robotics is a field that focuses on building and using robots. And guess what? AI plays a big role in robotics! Robots with AI can do more than just follow commands. They can learn from their surroundings, make decisions, and even interact with people. AI helps robots become smarter and more helpful in doing tasks, just like how our brains help us think and learn. It's really cool to see how technology is advancing and making robots more like us	Robots and Artificial Intelligence (AI) Learners need to acknowledge that AI and robotics work together to create intelligent robots that can do things on their own and adapt to different situations.!
R.5 Design a simple artefact based on a set of design instructions	
Project – Design and create your very own favourite animal prototype.     Use computational thinking, design thinking and the engineering design process to plan and design your animal prototype as follows: • Use reusable materials and a microcontroller and bring your animal character to life by programming its movement and incorporating sound. • Securely attach a microcontroller to control the movement and incorporate sound. • Link to R.7 to code the instructions for the animal character	

Content (Grade 5 / Term 4)	Notes/Examples
 	Learner code smiley faces using the microcontroller with a block-based coding environment. If it is possible to download the code to the physical micro controller, they can do so and create 'robots' Or they can stick a paper robot to the screen around the virtual micro controller (on screen) to create similar characters.
R.6 Mimic the operations of a robot	Link to C.7, R.5 and R.7
Example activity 1 – Solve problem using code blocks provided Refer to the activity in C.7 (moving through the maze). Someone coded the activity in a block-based coding application but did not complete the code. Open the activity Maze2.sb3, look at the algorithm in C.7, Consider the pattern, study the code provided and complete the code for the robot to move through the maze. The instructions are provided. Arrange them in the correct order. Test your program and debug until it works correctly. Learners need to refer to the steps outlined for coding a robot to perform a task (R.7 term 2)  **Example activity 2 - Flip-a-coin simulator** Write code to do the following: The microcontroller must simulate flip-a-coin. Decide which picture you want to display for heads and for tails. Possible solution: Choose a random number between 0 and 1. If 0, display "head" if 1, display "tail".)	Example activity 1 is an example of using the concept of Parsons puzzles – Refer to Grade 4 Term 1 C.3 The concept of Parsons puzzles is a type of scaffolded program construction tasks where the learner is given a set of code blocks of a single or multiple lines of code, and the task is to piece together a program from these or to fill in missing code from these. It helps learners to develop logical thinking.

Content (Grade 5 / Term 4)	Notes/Examples
R.7 Create, test and execute a set of robotic instructions	**Link to R.5 and R.6**
Example activity 1 – The purpose of this animation is to let a pet scroll across the screen. Display an image of a "pet" or object that can move (e.g. person, car) on the microcontroller. When you press button A the image must scroll from left to the right, until the image is not visible anymore. When you press button B the image must scroll from right to left, until the image is not visible anymore. (Tip: Display an updated "grid" each time). **Example activity 2** Code the animal character, execute and test the code and debug to fix if required. Execute the code to mimic the operations of your animal character (robot)	Do with R.5 and R.6
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.3, D.4. and D.5.**
Case Study: "Tech Tales - Exploring the World of Information Technology" **Introduction:** A group of grade 5 learners who embark on a virtual journey to discover the purpose of information technology, the fascinating world of computing devices, and the components of an ICT (Information and Communication Technology) system. Through their exploration, they will learn how technology impacts our lives and shapes the way we work and interact with the world. **Characters:** Emma: A curious and tech-savvy student who loves exploring new gadgets and learning about technology. Liam: A creative problem solver who enjoys using computers for drawing and designing. Olivia: A book lover who is excited to learn how technology helps in research and sharing information. Ethan: A sports enthusiast who wonders how technology is used in sports and fitness. **Scenario**: One day, Emma, Liam, Olivia, and Ethan visit a state-of-the-art technology exhibition held at their school. They encounter various interactive displays and activities that introduce them to the world of information technology and its many wonders. Activities and Discoveries: ***The Purpose of Information Technology:*** Emma learns about the primary purpose of information technology, which is to process, store, and transmit information efficiently. Liam discovers how technology empowers individuals to be creative and use computers for various tasks, like graphic design and animation. Olivia finds out how information technology aids in research, enabling access to vast amounts of information and knowledge. Ethan learns how technology enhances sports and fitness through wearable devices, training tools, and data analysis. ***Exploring Computing Devices:*** The learners interact with different computing devices, such as laptops, tablets, and smartphones. They learn how each device has unique features and functions designed for specific purposes. The learners discuss the advantages and disadvantages of different computing devices and how they impact daily life and learning. ***Components of an ICT System:*** At another exhibit, the learners discover the essential components of an ICT system. They learn about hardware components like the CPU, RAM, storage devices, and input/output devices (keyboard, mouse, monitor). The learners also explore software components, such as the operating system and applications, and how they work together to perform tasks.	**D.1, D.3, D.4 and D.5 done together** Reinforce and extend the following concepts: Technology → Information Technology (IT)→ ICT Computing devices **Possible discussion questions for case study** • What is the purpose of information technology, and how does it impact different aspects of our lives? • How do computing devices differ in terms of size, features, and functions, and how do they help us in various tasks? • Can you identify any instances where information technology has played a role in improving sports performance or health and fitness?
D.2 Recognise that he or she is living as citizens in a digital world.	**Link to D.6**
Example activity: Case study on digital citizenship **Scenario**: In a 5th-grade classroom, Alex, a tech-savvy student, discovers a funny meme on social media featuring a classmate, Emma. Excited, Alex screenshots it and shares it in a group chat. Unaware of Emma's feelings, she finds the meme hurtful and embarrassing, as it was shared without her consent. Guiding Principles for Responsible Use of Technology (to be discussed with learners): • Care for your device	Possible discussion Questions for case study: • What did Alex do in this situation? Was it a responsible use of technology? Why or why not? • How do you think Emma felt when she saw the meme? How could this have affected her emotionally?

Content (Grade 5 / Term 4)	Notes/Examples
• Respect Others: Always think about how your actions might affect others before sharing or posting anything online. Treat others with kindness and respect in your digital interactions, just as you would in person. • Think Before You Post: Take a moment to consider the potential consequences of what you are about to share online. Ask yourself if it is appropriate, kind, and if you have the permission to share it. • Seek Consent: Before using someone else's photos, videos, or personal information, always ask for their permission. Respect their decision, even if they decline. • Be Empathetic: Put yourself in others' shoes and try to understand how they might feel about what you're sharing or posting. Avoid anything that might hurt or embarrass others. • Report Inappropriate Content: If you come across inappropriate or hurtful content online, report it to a trusted adult or a teacher immediately. Do your part in helping maintain a safe online environment. • Report inappropriate behaviour	• Can you think of any ways in which Alex could have used technology more responsibly in this situation? • What are some potential consequences of sharing inappropriate content or memes without permission? • How could this situation have been handled differently to ensure a positive online environment for everyone?
D.3 Demonstrate an understanding of the concept of a computing device.	**Refer to D.1** **Done with D.1**
D.4 Identify the common uses of ICT in the real world	
D.5 Differentiate between the components of an ICT system	
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.2**
Case Study: "Tech Transformations - How Technology Reshaped Our World" In this case study, we will explore how the adaptation of technology has significantly impacted the world we work and live in. We will follow the journey of a town called Techville and its inhabitants, as they experience various technological advancements and their effects on their daily lives and work. ***Characters:*** Sarah: A grade 5 student living in Techville, curious about the changes brought by technology. Mr. Khumalo: A shop owner in Techville who has witnessed the evolution of technology in the business world. Mrs. Mbhele: A teacher who has experienced the transformation of education through technology. Mr. Johnson: A farmer in Techville who embraces modern agricultural technology. Scenario: Techville was once a small town with limited access to technology. Over the years, technology has gradually found its way into the lives of its residents, revolutionizing the way they work and live. Activities and Discoveries:  ***The Early Days:*** Sarah interviews Mr. Khumalo, who has been running a family-owned shop for decades. Mr. Khumalo shares how he used to manage inventory manually and the challenges he faced in reaching customers. Sarah learns about the impact of the internet and e-commerce on Mr. Khumalo's business, allowing him to expand his customer base and streamline operations. ***Education in the Digital Age:*** Sarah talks to Mrs. Mbhele, a long-time teacher, about the changes in education due to technology. Mrs. Mbhele explains how traditional teaching methods have evolved with the introduction of digital tools and online resources. Sarah discovers how technology has enriched learning experiences and made education more accessible to learners. ***Revolutionizing Agriculture:*** Sarah visits Mr. Johnson's farm, where she witnesses the use of advanced agricultural technology. Mr. Johnson demonstrates how modern farming tools, such as tractors with GPS, help optimize crop yield and reduce manual labour. Sarah learns about the importance of precision agriculture and its positive impact on sustainable farming practices.	Possible discussion questions for case study: • How has technology transformed Mr. Anderson's shop, and what benefits has he experienced from embracing e-commerce? • How has technology enhanced Mrs. Mbhele's teaching methods, and what role does it play in enriching learners' learning experiences? • What are the advantages of modern agricultural technology, as demonstrated by Mr. Johnson, and how does it contribute to sustainable farming practices?
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.1**
Part of case study with D.1.	Done with D.1

Content (Grade 5 / Term 4)	Notes/Examples
D. 8 Interpret a pattern to represent or communicate a message or image	**Link to D.9 and C.2 – C.5 and R.5 – R.7**
Provide learners with a ‘pattern’/rules/algorithm for decrypting a message which they need to interpret and figure out. Explain to them that the same pattern for encrypting a message is used for decrypting the message. Act like a detective! **Example activity 1** Agent Sipho and Agent Alice send each other encrypted messages using the following algorithm.  *https://olympiad.org.za/talent-search/past-papers/pen-and-paper/* Study the encryption algorithm and write the algorithm to decrypt the message. Agent Alice receives the encrypted message “PMGEP" from Agent Sipho. Use your decryption algorithm to decode the message and write down the decoded message.	Codes and ciphers are forms of secret communication. A code replaces words, phrases, or sentences with groups of letters or numbers, while a cipher rearranges letters or uses substitutes to disguise the message. Learners need to know, at a basic level, that Sensitive/confidential information on the internet such as credit card numbers and passwords are encrypted using various encryption ‘rules’/programs (software)
D.9 Create a pattern to represent or communicate a message or image	**Link to D.8 and C.2 and C.5 and R.5- R.7**
Learners use block-based coding applications to interpret and communicate messages using text/images/LEDs/interactive stories, etc.	**D.8 and D.9 can be done together**
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to Cs and Rs**
Revise and extend from previous grades and terms. Learners use Paint to create sprites and backgrounds for their block-based coding applications to import into their block-based coding applications Learners practise file and folder management	

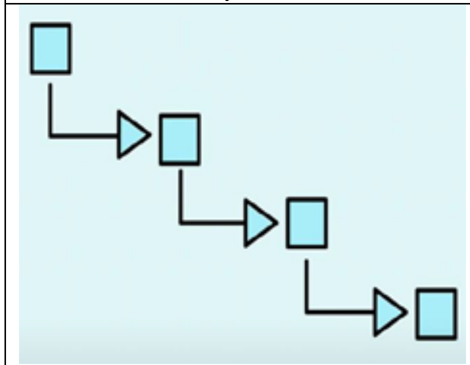
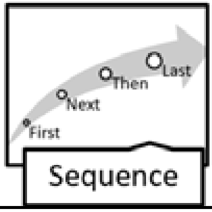
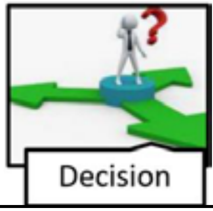
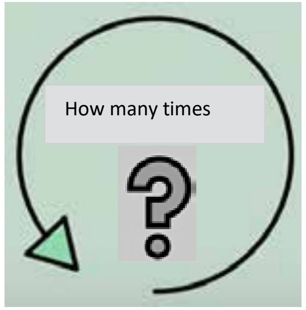
3.3 Grade 6

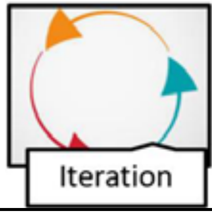
Note:
Teachers must include the following competencies and content in their Annual Teaching Plan (ATP), distributed across the terms and sequenced, organised and grouped in a manner that will facilitate learning in a manner that will make sense for learning and teaching, maximize the learners' learning outcomes and achievement. and in a way that will make optimal use of time and resources. Some competencies could also be combined in bigger/more complex programs.

Also, in Grade 6, it is advisable to complete all the Coding in Term 1 and Term 2 and then complete the Robotics in Term 3 and Term 4. Digital concepts are spread across the four terms as it supports both Coding and Robotics.

3.3.1 Term 1

Content (Grade 6 / Term 1)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.2 – C.7, R.5 – R.7**
Example activity 1 – Follow a set of instructions: Provide learners with the instructions below. They must the following instructions *exactly* to draw a picture. 1. Draw a diagonal line. 2. Draw another diagonal line connected to the top of the first one. 3. Draw a straight line from the point where the diagonal lines meet. 4. Draw a horizontal line over the straight line. 5. At the bottom of the straight line, draw a curvy line. 6. Draw a diagonal line from the bottom of the first diagonal to the straight line. 7. Draw a diagonal line from the bottom of the second diagonal line to the straight line. This activity illustrates the importance of sequence and detail. It is important that learners follow the instructions precisely as they interpret. They cannot ask for help, and they may not look at a friend's work (they must do it themselves) When done, ask some learners to show their drawings and compare the drawings, then ask learners to answer the following questions: • Are they different? • Why? • What was difficult about following the instructions? • What was missing from the instructions? Now tell them that the drawing was supposed to be a kite and ask them to write a set of improved instructions that that someone could follow to draw the picture. They must make sure that: • There is only one way to interpret each step, that is, all instructions are unambiguous. • To break down (decompose) instructions where required. • To provide enough detail in each step • That the instructions are in the correct order Now, in pairs or small groups, let them write down the characteristics of a good algorithm. **Example activity 2** You need to explain to someone that is using WhatsApp for the first time how to send a WhatsApp message. You found the following instructions to send a WhatsApp message: • Type message • Open WhatsApp • Send message	Both **sequence** and **detail** are important when developing an algorithm Attention to **detail** is also important as it helps prevent mistakes and ensures successful completion of a task. ***Detail*** means considering every aspect or minor part of something. It is to describe or give ***exact*** information about something. The steps or instructions to perform a task also need to be ***unambiguous*** – they need to be ***precise*** and clear to avoid misinterpretation or different interpretations by different people. **An Algorithm** is a set of well-defined steps or instructions that are followed to perform a specific task or solve a particular problem. The instruction set can be sequential or can include branching (decision structure) or repetition (loops). **Key characteristics of a good algorithm: Each step** • must be clear and unambiguous. • must be at the right level of **detail** and specific. • consists of a single task (be at the most basic level) • must be in the correct, logical **sequence** • must be correct/solve the problem **Remember** One uses CT in all tasks that one wants to complete appropriately.

<table>
<tr><th>Content (Grade 6 / Term 1)</th><th>Notes/Examples</th></tr>
<tr><td>Rewrite the above instructions to include more detail/steps to make them more precise so that anyone that follows the steps will exactly know what to do and be able to perform the task successfully. Then hand your instructions to a friend to check your instructions for sequence and detail.</td><td>It helps one to approach problems more systematically and develop well-structured solutions. or find an efficient an effective solution to a problem.</td></tr>
<tr><td>C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.</td><td>Link to C..1 and C.3 – C.7</td></tr>
<tr><td>By now learners need to understand the following concepts
<table>
<tr><th>Sequence</th><th>Selection</th><th>Repetition</th></tr>
<tr><td>
A series of actions are performed in a specific order, the first action first, then the second action, then the next action until the last action.

</td><td>
There is sometimes more than one path to follow, and we need to ask a question to decide which path to follow

</td><td>

Loops also ask questions, such as how many times? but ask the question over-and-over again and perform actions over-and-over until the condition is satisfied

</td></tr>
</table></td><td>Note
Sequence –Learners must now be able to write basic code in the correct logical sequence.

Learners must now be able to use the following control structures when completing simple tasks/solving simple problems:

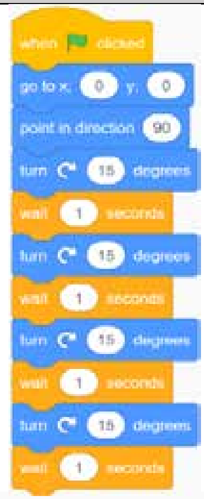
Selection – learners must now be able to do a basic
IF…THEN
IF….THEN….ELSE
Based on a simple condition

Repetition – learners must be able to use a
• Forever loop
• Repeat loop with a fixed number (constant) of iterations

Note
Programming concepts are abstract and intermediate phase learners still didn't reach the formal stage of cognitive development. Therefore, abstract thinking is still not reached.</td></tr>
</table>

<table>
<tr><th>Content (Grade 6 / Term 1)</th><th>Notes/Examples</th></tr>
<tr><td>

Example activity 1 – two sprites, same action
Provide learners with the code on the right (two sprites have the same code)
Let them execute the code and watch what happens.
Learners now explain the code.
Now, learners write their own code for two sprites to interact, using their knowledge, skills and experience gained thus far



</td><td>

Note:
It is important that coding activities revise coding concepts learned in previous terms and grades cumulatively.

</td></tr>
<tr><td>

Example activity 2 – Length of word/string and join multiple words/characters
Learners study the code on the right and write down what the output would be.
Then they explain what the program does.

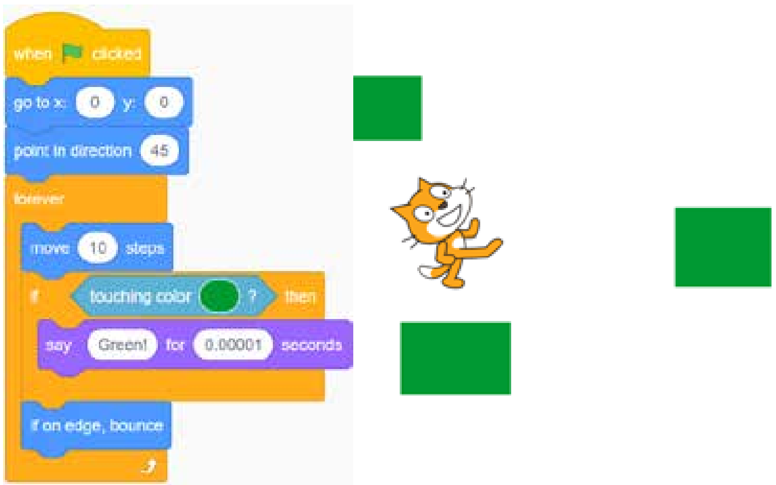
</td><td>

Note:
Learners generally struggle to stack an output that uses multiple joins correctly.

</td></tr>
<tr><td>

Example activity 3 – practise multiple joins
Provide learners with activities to practise multiple joins.

Example activity 4 – Forever touching colour
On a worksheet, provide learners with the code on the right.
Let them study the code and describe what the program does.
Now, let them implement the code in a block-based coding environment and compare their description with the outcome after executing the code.



Example activity 5 – Open ended
Leaners use their knowledge, skills and experience to write a program of their choice that uses similar code and some other concepts that they have learned so for

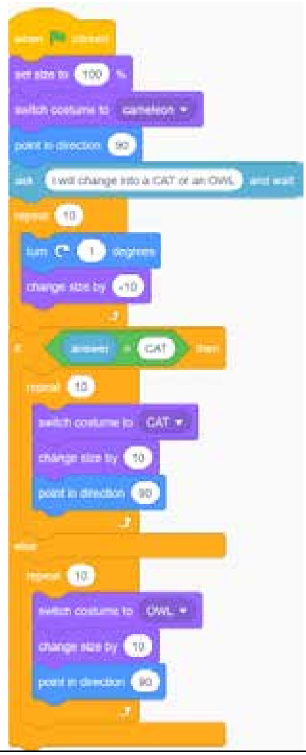
</td><td>

Note:
While learners should be able to describe what each line (block) of code does, (describing a code segment line-by-line/block-by-block) it is important that learners explain the overall purpose of the code, i.e. what the program does/the purpose of the program.

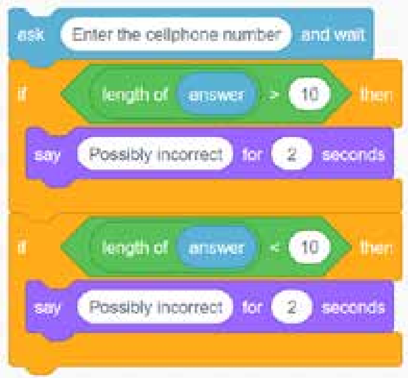
Remember
An Algorithm is a set of well-defined steps or instructions that are followed to perform a specific task or solve a particular problem. The instruction set can be sequential or can include branching (decision structure) or repetition (loops).
Key characteristics of a good algorithm: Each step

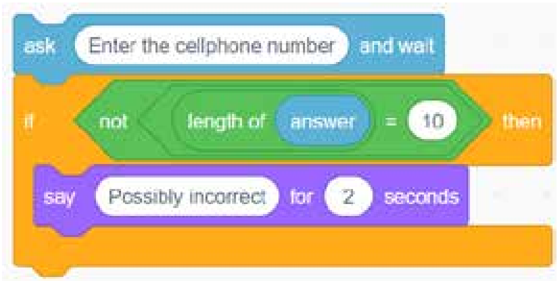
- must be clear and unambiguous.
- must be at the right level of ***detail*** and specific.
- consists of a single task (be at the most basic level)
- must be in the correct, logical ***sequence***
- must be correct/solve the problem

</td></tr>
</table>

Content (Grade 6 / Term 1)	Notes/Examples
C.3 Interpret and execute a given symbolic or written set of commands	
Example activity 1 Backdrop changes while music plays (graphic effects and brightness) Run the code on the left, then study the code and explain what it does,   **Example activity 2** Write code to implement effects like that of example activity 1. Combine these effects with code of your choice. **Example activity 3 – Introduce IF…THEN..ELSE** Revise the IF…THEN structure Now, demonstrate the program in the right to learners. Draw their attention to the IF…THEN…ELSE structure Discuss the structure does Provide learners with the code to study  **Example activity 4 - Consolidation** Teacher provides learners with a task/problem that uses an IF…THEN…ELSE structure and some other code that they learned so far, which they need to plan, code, execute, test and debug. **Example activity 5 – Open ended** Leaners use their knowledge, skills and experience to write a program of their choice that uses a variable	**Note:** Provide learner with activities enabling them to • read code and explain what it does or • work through (trace) / act out code (physically or simulated) to determine the output or the correctness or • provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or • translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions)) • add some functionality/instructions to an existing program. • rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or • choose the correct solution from 2-3 options or • compare different solutions to evaluate efficiency or • debug an algorithm or block-based program (find the bug, describe the bug and correct it) • develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging. **Note**: Evidence suggests that pupils should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practice (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).

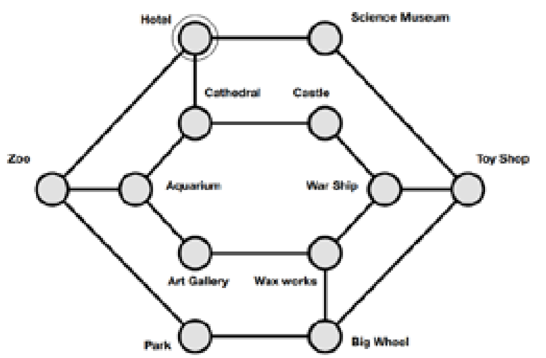
<table>
<tr><th>Content (Grade 6 / Term 1)</th><th>Notes/Examples</th></tr>
<tr><td>C.4 Debug a given symbolic or written set of instructions</td><td></td></tr>
<tr><td>Provide learners with a program where an error that leads to incorrect output. (Tell them what the output should be)
Learners need to study the code and correct the output.</td><td>Though debugging is part and parcel of coding, provide learners with code where a deliberate error was made, and let them find and correct the error.</td></tr>
<tr><td>C.5 Evaluate a given solution towards potential improvement</td><td></td></tr>
<tr><td>Example activity
Kanthan and Sipo both wrote a program to evaluate the possible correctness of 10 cellphone numbers (a cellphone number is possibly correct if it contains 10 digits)
Evaluate their respective programs and determine which code is better. Explain why.





Both the programmes only test for one number. Complete and improve the code to achieve the outcome (test 10 numbers).</td><td>Learners also need to evaluate code, e.g. two programs with the same outcome but different approaches in code.
They need to understand that there can be more than one solution to a problem, however, some solutions are better/more efficient than others.

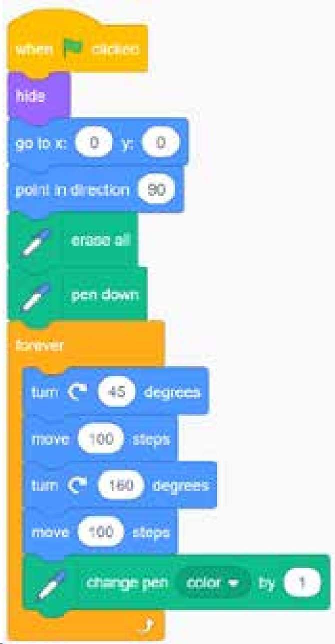
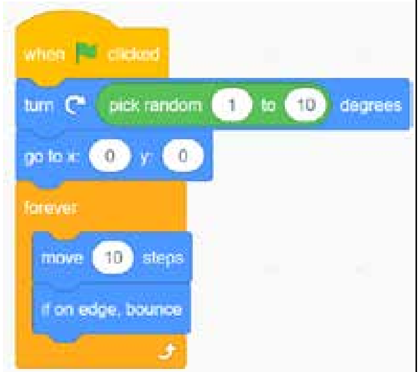
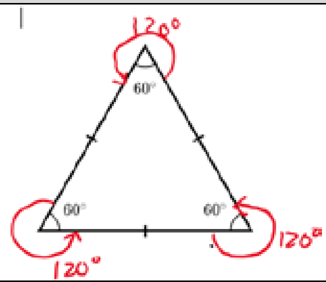
Note:
Literature suggests that the biggest problem of novice programmers does not seem to be the understanding of basic coding concepts but rather learning to apply them. Therefore, at this level, beware of giving learners programming tasks that combine too many concepts (Robins, 2019).</td></tr>
<tr><td>C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.</td><td></td></tr>
<tr><td>Example activity 1 Maps and routes
You are a hotel tour guide. Tourists staying in your hotel expect to be taken on a tour visiting all the city's attractions. You have been given an underground map that shows all the locations of the attractions and how you can get from one to another using the underground network. You must work out a route that starts from the hotel and takes your tour group to every tourist site. The tourists will be unhappy if they pass through the same place twice. They also want to end up back at their hotel that evening
cs4fnpuzzlebook11.pdf (wordpress.com)



Example activity 2 – Bouncing ball
Write code for the following algorithm:
Turn right in a random direction (degrees)
Go to the middle of the stage.
Forever
    Move 10 steps
    If on edge bounce
Execute the code, test, and debug until the required outcome is achieved.

Explain to learners that, in the algorithm the indented instructions below the Forever instruction implies that the indented instructions are part of the loop.</td><td>Note:
Concrete activities remain important as literature suggests that the primary weakness of today's pedagogy of programming is that it doesn't provide enough opportunity for the novice to develop concrete operational skills, via the correct types of exercises…due to too much emphasis on writing large amounts of code, and problem solving.

By identifying patterns, we can predict what will come next and what will happen again and again in the same way.
In Computer Science/coding we analyse patterns in data and make predictions and generalisations based on the pattern analysis.</td></tr>
</table>

Content (Grade 6 / Term 1)	Notes/Examples
Example activity 3- Forever circle pattern Learners study the code on the right, execute the code and explain what it does. Learners then change some of the values and study the effect. **Example activity 4 – Open ended** Leaners use their knowledge, skills and experience to write a program of their choice using what they have learned. 	Possible solution for activity 2: 
C.7 Create or complete a pattern to represent a data set	
Example activity 1 – Draw triangle Provide learners with the following drawings: Requestion leaners use their knowledge, skills and experience from drawing a square and a rectangle in previous grades to draw an equilateral (all sides are of equal length and all angles are the same size) triangle. They need to understand that the sprite turns on right, i.e. on the outside edge (120 degrees). Let them explain how the instructions for drawing a triangle differ from the instructions for drawing a square? 	Learner now builds on drawing a shape from Grade 4 and Grade 5 (where they coded a square and rectangle)
Content (Grade 6 / Term 1)	**Notes/Examples**
Robotics	
R.1 Explain what a robot is in simple terms.	**Link to R.2 and R.3**
R.2 Identify different types of robots.	**R.1 – R.3 done together** Revise and extend to include the following sensors and their purpose • Vision sensors (cameras) enable the robot to "see" and recognize objects, colours, and shapes. • Infrared sensors help the robot detect proximity or measure temperature, allowing it to navigate and avoid obstacles.
R.3 Outline the different components of a robot	
Example activity 1 Briefly revise what a robot is, different types of robots and the main components of a robot from previous grades. Extend by showing learners the picture below an discuss the different sensors that robots could have.	

Content (Grade 6 / Term 1)	Notes/Examples

(https://www.researchgate.net/figure/Mobile-robot-sensor-systems_fig1_221909161)

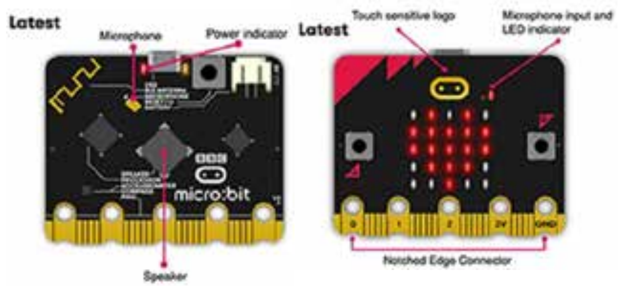
Notes/Examples

- Sound sensors (microphones) allow the robot to perceive and interpret audio cues, such as speech or specific sounds.
- Ultrasonic sensors assist the robot in measuring distances and detecting obstacles for effective navigation and obstacle avoidance.
- Motion sensors can trigger specific actions or alert the robot to the presence of movement in its surroundings.
- Wheel encoders provide precise information about the robot's movement, speed, and distance travelled, aiding in accurate navigation and control.

By using a combination of these sensors, robots can gather a comprehensive understanding of their environment, make informed decisions, and perform tasks more efficiently.
https://www.futurelearn.com/info/courses/robotic-future/0/steps/29367

R.5 Design a simple artefact based on a set of design instructions	Link to R.6 and R.7

Example activity 1 – Introduce the physical microcontroller.
Explore physical computing using the microcontroller device; discuss and demonstrate out how it can be used to develop understanding of programming through creative digital projects. exploring the in-built buttons, LEDs, and sensors for detecting movement, sound, light and heat.
Show learners how to setup the device.
The following video could be used to introduce learners to physical computing:
https://youtu.be/X3zxNSIFsdQ



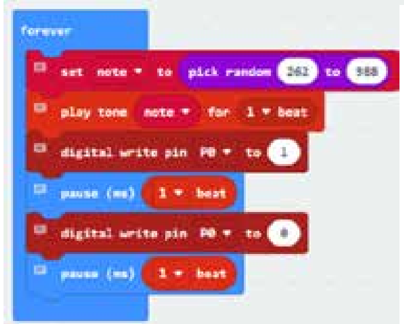
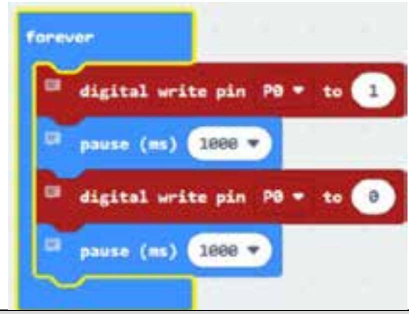
Example activity 2 – Blinking LED
Learners use the microcontroller to control an LED, making it blink in a continuous loop.

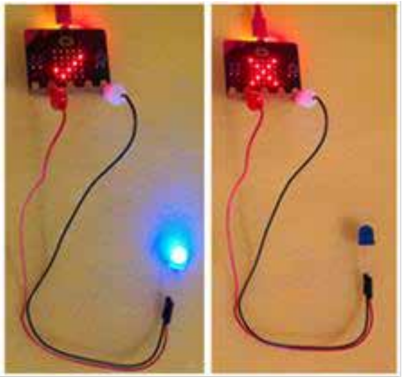
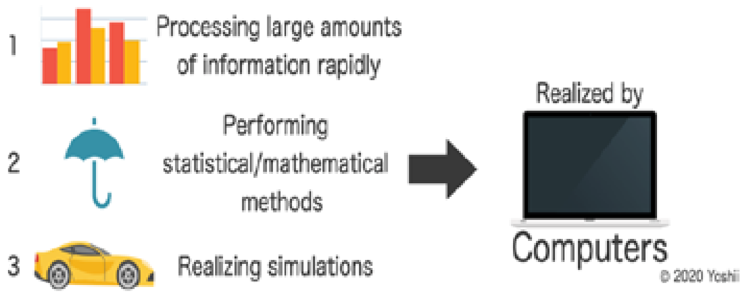
Notes/Examples

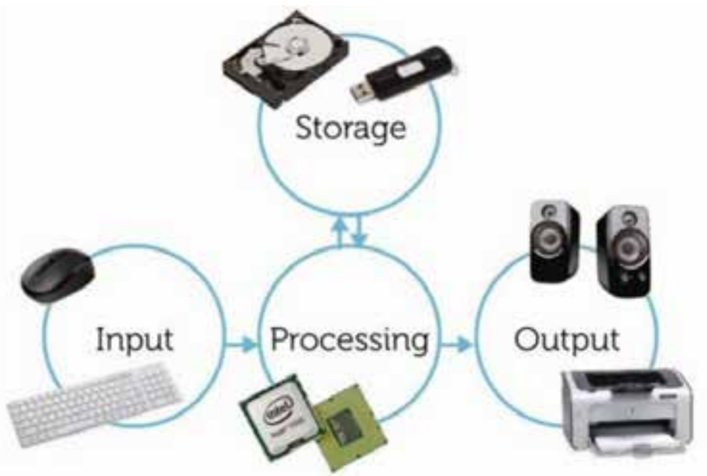
R.5, R.6 and R.7 are done together in Term 1, using basic activities

Learners are introduced to physical computing.
They need to explore the physical microcontroller and link this to the virtual microcontroller that they experienced in Grade 5.
Learners are gradually exposed to the built-in buttons, LEDs and sensors for movement, sound, light and heat through various activities

Learners need to know
- what an LED is and
- how it can be controlled using digital signals.
- The concepts of digital and analogue input

Content (Grade 6 / Term 1)	Notes/Examples
Example activity 3 – Use light sensor on microcontroller to control and LED Introduce learners to the concept of analogue inputs, IF-THEN-ELSE statements, and the use of sensors in programming Show how the LED turns on when the room is dark and off when the room is light. Explain that this is done using an IF-THEN-ELSE statement in the code.  **Example activity 4 – LED music visualiser** Use the microcontroller to play a melody and display a light show on an LED at the same time. This show learners to the concept of random numbers, simultaneous execution of tasks, and the use of music in programming.  **Example activity 5 – LEDs blink in a continuous loop** Use microcontroller to demonstrate a forever-loop with blinking LEDs 	**Note**: Coding concepts and principles are learned in the coding section and need to be transferred to the robotics block-based environment. **Note**: Evidence suggests that pupils should be taught – initially at least – in small bite-sized chunks. These steps in the learning process should be well-thought out and gradual as well as allow plenty of opportunity for practice (see, for example, Rosenshine, 2012; Coe *et al*., 2014; Sealy, 2019).
R.6 Mimic the operations of a robot	**Link to R.5 and Cs**
R.7 Create, test and execute a set of robotic instructions	R. 6 and R.7 are done together with R.5 (once enabling activities in R.5 are completed) to complete the project. Learners need to demonstrate how • LEDs work • to connect them to a microcontroller and • how to program the LEDs to change colours after a specific time, simulating the 'stop' and 'go' signals of a pedestrian crossing
Project - Design and Create a Pedestrian Crossing using Microcontrollers and LEDs	

Content (Grade 6 / Term 1)	Notes/Examples
Use computational thinking, design thinking and the engineering design process plan, develop, execute and test a set of instructions to simulate a pedestrian crossing 	
Digital Concepts	
D.1 Outline the concept of technology and purpose of information technology (IT)	**Link to D.3, and D.7 and C.2 and R.5**
Learners relate the concept of IT to • ways for processing large amounts of data and information rapidly, e.g., supermarkets need to process tons of sales every day such as calculating the total amount of sales using software (spreadsheets, sales programs) that enables one to just click a button to see daily sales and breakdowns of the sales, etc. • the application of statistical and mathematical methods to decision-making, e.g., if the weather forecast tells us that there is an 80% probability of rain, we decide to take an umbrella when we go out. • the simulation of higher-order thinking through computer programs such as a racing video game that simulates the behaviour of a car using computing devices that enable the above What Is Information Technology? ~ 3 Little-known Definitions of IT ~ | Yoshi's IT (yoshi-it.com)	Done with D.3 and D.7 Learners need to understand that • Information Technology (IT) specifically refers to the use of computers and software to manage and process data and information. • The purpose of Information Technology (IT) is to use computers, software, and other technology tools to manage, process, store, and present information in various contexts. • In the context of information technology, a computer is an electronic device designed to process, store, and retrieve data through various programs and applications.
D.2 Recognise that he or she is living as citizens in a digital world.	Link to D.6 and D.3
Provide a simple explanation of the digital world all around us and remind learners what digital citizenship means and that it can be described as the quality of habits, actions and consumption patterns that impact the ecology of digital content and communities. Explain how to use technology and computers in the classroom responsibly Focus on caring for the computing devices. **Example activity** Learners engage in an activity that let them understand: Just like we take care of our precious physical possessions, we also need to take care of the computing devices we use in the classroom to ensure their longevity and to contribute to a safer and more sustainable digital environment. Focus on aspects such as physical care, secure storage, etc. 	Done with D.6 **The Digital World**: The digital world is a vast and interconnected realm of information and communication that surrounds us in our daily lives. It encompasses all the digital technologies and networks that enable us to access information, communicate with others, and interact with various digital platforms. **Digital Citizenship**: Digital citizenship refers to the responsible and ethical use of technology, particularly in the online space. It involves using digital tools, devices, and platforms in a way that respects the rights and privacy of others.

Content (Grade 6 / Term 1)	Notes/Examples
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.1, D.4 and D.7, C.1 – C.5 and R.5 – R.7**
Example activity 1 – Reinforce and extend concepts previously learned. Provide learners with a diagram of a computing device. Learners describe and give examples of • Computing device • Input • Output • Processing • Storage  A computer is an electronic device that processes data and performs various tasks according to a set of instructions provided through software or programs. It consists of hardware components that work together to enable users to interact with software, access information from the internet, create documents, and perform countless other activities. ***Input***: Device or component that allows information to be given to a computer (e.g., keyboard, mouse, touch screen, sensors, etc) ***Output***: Device or component that receives information from the computer ***Processing***: Steps that are done with the information, e.g., calculating, sorting, etc. Introduce the microcontroller as computing device **Example activity 2:** The microcontroller as a computing device Use the microcontroller and let learners describe the input, e.g., sensors, and buttons on the device. Allow them to identify the output from the microcontroller, e.g., speaker. Ask them which app can be used with the device and the function of the app. Learners watch the following videos: https://youtu.be/Y9tk07CzTAA (processor and https://youtu.be/NkoS2JXaBuM (input and output) **Input:** the data or signals that a device receives from its surroundings. In the context of the microcontroller, input can come from various sensors and external components. The microcontroller has built-in sensors and buttons that serve as sources of input **Processing**: involves taking the input data, performing calculations or operations on it, and making decisions based on that data. **Output**: Refers to the results or actions produced by processing the input data. In the case of the microcontroller output is typically displayed on its LED matrix or heard through its built-in speaker.	Learners need to • Explain what a computing device, is in terms of input, processing, output, and storage (in the context of IT). • List common input, output, and storage devices. • Explain the purpose and role of hardware (as input, processing, storage, and output devices) and software as a list of instructions (apps) that the computer can follow'. Extend the concept of a computing device to the microcontroller (See D.7) Learners need to understand that the microcontroller is a small computer that can receive input, do some processing and provide output, though, e.g., buttons, pins and sensors. These inputs are processed and provide output through e.g., LEDs
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.2**
The technological ease of copying, pasting, clicking and sharing content online has given rise to the fast spreading of false/fake news.  **Example activity: Unravelling False/Incorrect Information/Fake news.** Learners watch the following videos: https://youtu.be/D0Cd9-eJ-No and https://youtu.be/xDLohXNgF4o Provide learners with a worksheet with the following questions: • What is false information? /Fake news? • Why do people spread false information? • Is it a matter-of-fact vs opinion? • Is it sponsored stories disguised as news on social media?	Learners need to understand that, however, they have access to information they must be aware that anyone can post information on the internet of distribute information via social media, so they need to be vigilant and able to identify incorrect/false information or fake news. **Fake news/False information** News or stories on the internet that are not true. They may be in the form of disinformation or misinformation. **Disinformation** False information that's created and shared to deliberately cause harm.

Content (Grade 6 / Term 1)	Notes/Examples
• Is it deliberate to achieve some or other goal? • What potential harm could be caused by allowing false information to stand uncorrected? Learners could also use KWLS chart with the above activity	**Misinformation** Generally used to refer to misleading information created or disseminated without a deliberate intent to cause harm. Learners need to know: • What false information/fake news is • Why people spread false information/fake news. • What potential harm it can cause
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.1, D.3, and C.1 – C.5 and R.5 – R.7**
Example activity 1 input, processing and output between a computer and a microcontroller. **Example activity 2** Illustrate the concept of IPO using a simple real-life example like a recipe: • Input: Ingredients (e.g., flour, sugar, eggs) • Process: Mixing the ingredients and baking • Output: A delicious cake Emphasize that just like in cooking, computers follow the same input-process-output approach. Present a scenario where GIGO can cause problems with the recipe, e.g. too much flour or adding replacement ingredients, etc. What will happen with the output? (cake) Discuss how bad inputs can lead to inaccurate outputs and highlight the importance of providing accurate data when programming	**Done with D.1, D.3** Learners need to • Distinguish between input through instructions that are executed and results in action and output as a form of communication from the device. • Describe the interaction/relationship between input, processing, and output. (e.g., using algorithms/black-based coding) • Compare input, processing and output of computer and microcontroller • An elementary understanding of storage elsewhere (not on device e.g., cloud storage). • Know that incorrect input results in incorrect output (GIGO)
D.8 Interpret a pattern to represent or communicate a message or image	**Link to D.9 and C.1 – C.5 and R.5 – R.7**
D.9 Create a pattern to represent or communicate a message or image Communicate a message using encryption and decryption • Use a simple cipher to create (encrypt) and communicate a message or design an image (e.g., text to communicate a message) • Decrypt an encrypted message using the same cypher that was used to encrypt a message. • Communicate a message using images, e.g., LEDs on micro-controller (Do with R.5 – R.7).	**D.8 and D.9 done together** Learners need to understand: Encryption and decryption are sides of the same coin - one uses the same 'rules' to decrypt as what were used to encrypt a message.
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.2 – C.5 and R.5 – R.7**
<u>Example activity 1:</u> Creating a file structure and naming conventions for Grade 6 work (include purpose of and understanding of file extensions) Explain the concept of a file structure. Explain that a file structure is like a virtual filing cabinet that helps organize files and folders on a computer. Demonstrate how to create a simple file structure using a flowchart or diagram. For example: • Main Folder (e.g., "School Work") • Sub-Folders (e.g., "Math," "English," "Science") • Files (e.g., Programming. LED_light_Program.bbb ") Discuss the benefits of using a file structure, such as easy access, efficient searching, and avoiding clutter. Emphasise good file and folder naming conventions. **Example activity 2 – File management** Distribute worksheets with partially organized file structures and incomplete file names. Learners need to • complete the file structures and rename the files following the naming conventions discussed. • Write down the file path to specific files	Revise and extend the following competencies: • Load/open, save, and run a block-based coding application. • Explain the purpose of a file extension. • Explain and demonstrate the concept of saving files using a descriptive filename and file extension. • Create and name a simple folder structure for saving files. • Explain file and storage management – basic file management. • Save and Open filles from within an application as well as following a file path. • Design a simple sprite and a simple backdrop using an application such as Paint to import and use in a block-based application. • Design a customised 'GUI' for a block-based application

3.3.2 Term 2

Content (Grade 6 / Term 2) | **Notes/Examples**

C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.

Link to C.2 – C.7 and R.5 – R.7

Example activity CT in real life
Using computational thinking, learners plan their birthday party:
Break down the most important aspects of a birthday party (main ideas – abstraction + decomposition) in terms of tasks and timeframes, e.g. 1 month before determining final date and time for the party, etc.
Develop a programme (algorithm – step-by-step guide) for the day – what will happen when and in which order, etc.

Using their experience of party planning and abstracting the important information, identify patterns such as always start with identifying the date, budget, the guest list, theme and venue.
Break down organising the party into steps, asking questions by first look at the pattern such as the order of the tasks and the time frame and decompose tasks, e.g.

- 1 month before:
  - When? Budget? Theme? Where?
- 3 weeks before
  - Book venue
  - Who will I invite? (Guest list)
  - Design an invitation card
  - Distributing the invitations to all names on the guest list
- 2 weeks before
  -
- 1 week before, etc

Etc.

C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.

Link to C.1, C.3-C.7 and R.5 – R.7

Example activity 1 – Introduce variables
Explain to learners what a variable is and how to create a variable.
Like each strawberry can only have ONE size, a variable can only hold one value at a time.

Example activity 1:

- If Bafana Bafana and Brazil play a soccer game, before the game starts, the scoreboard shows the value of 2 variables, i.e., ***scoreBafana*** and ***scoreBrazi***l as indicated on the right. As none of the teams has scored the values for both are 0
- If, after 10 minutes Bafana Bafana scores a goal, the scoreboard changes as indicated on the right (The value of the variable ***scoreBafana*** now changes to 1, but the value for ***scoreBrazil*** remains 0
- After another 15 minutes Bafana Bafana scores another goal, and the scoreboard changes again (The value of the variable ***scoreBafana*** now changes to 2 as indicated on the right (the previous score (1) is overwritten)
- Just 5 minutes before the final whistle Brazil scores a goal and once more the scoreboard changes to reflect the latest score (The value of the variable ***scoreBrazil*** now changes to 1 as indicated on the right:

scoreBafana	***scoreBrazil***
0	0

scoreBafana	scoreBrazil
1	0

scoreBafana	scoreBrazil
2	0

scoreBafana	scoreBrazil
2	1

Like the values of the variables called ***scoreBafana*** and ***scoreBrazil*** that change when one of the teams scores a goal and can only contain one value at any specific time during the game, a variable in a program can change through input by the user (like the Ask and Answer) or writing code to change the value of a variable.

In previous grades learners used **ANSWER** to keep a value to be used later in a program. However, ANSWER is a sensing block and a reporter block. It reports the most recent text/value inputted with the ***Ask and wait block***.

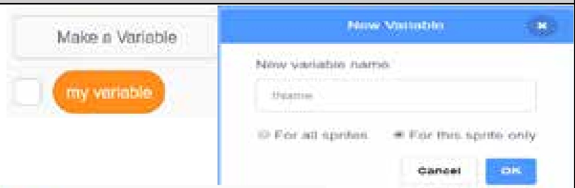
Introduce variables
Variables are needed to run all but the simplest computer programs. As a program runs, it needs to hold information in its memory. Variables allow us to store, change and access this information as the program runs or executes.
Imagine you are playing a game. Every time you win, you get a point that is added to your score.

- A variable is used to store the score (value).
- A variable can only store ONE value at a time
- The value of a variable can change
- We can change the value of a variable through code
- The value of the variable can be accessed throughout the program (unlike with answer that only provides the most recent input of the Ask block)
- A variable has a name and a value (we will not deal with type now)

Content (Grade 6 / Term 2)

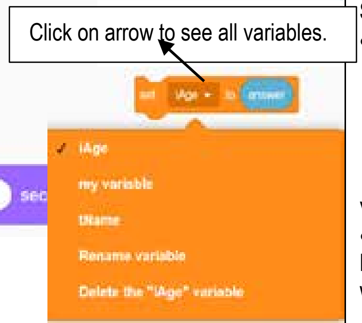
Example activity 2 – Create and use a variable

Select the "Variable" Blocks and the following will appear.
Click on ***Make a variable*** and type the variable name in the window that pops up

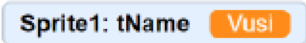


On the stage, in the left corner, you will see your variable (starting with a 0 – just like the soccer scoreboard)

Now write the code on the right, then run the program and watch what happens



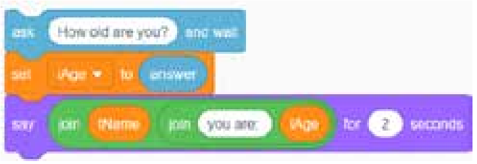
This example still works pretty much like the **Ask and Answer**, except for one difference:
The name is now stored in the variable, tName



(the 0 has been overwritten and the value of the variable is now Vusi)

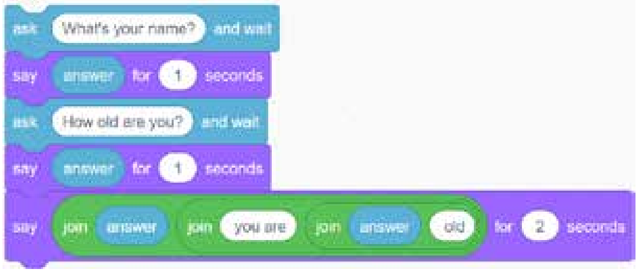
Now, let us add to the code on the right, but first make another variable called iAge
You will now see two variables at the top:

Can you already explain the differnce between only using Ask and Answer vs using variables?
Add the folllowing blocks and run the program



What would have happened if you did not use variables (only used **Ask and Answer?** What would then have been displayed?

See if your answer is correct by wrrting the following code then run it:



Can you explain the difference between using only Ask and Answer as opposed to using variables?

Now, explain what the **Set** instruction does

Notes/Examples

Note:

Consciousness about teaching variables to novices is important as many novices form misconceptions (incorrect mental models) of programming concepts, of which misconceptions about variables are common. If learners gain misconceptions about variables, it is likely that they will also struggle to master other programming concepts like branching or repeating (Zanko et al, 2019).
Some misconceptions include, e.g.:

- **Assignment** (*set…to (answer or other value)*), e.g.

Learners may believe that iNumber will contain the first assigned value (10)
Or maybe the sum of the two values (15)

- **Output** (e.g., *say…)*

Learners may believe that the variable name rather than its value will be outputted (say…)

Note:

As many learners tend to struggle with the concept of a variable, it is important that learners initially, practise in writing many small, basic programs using one variable with each program.

Initially learners should be explicitly instructed to declare and include variables as part of their programs. They should not be required to deduce that a variable is required to solve a problem.

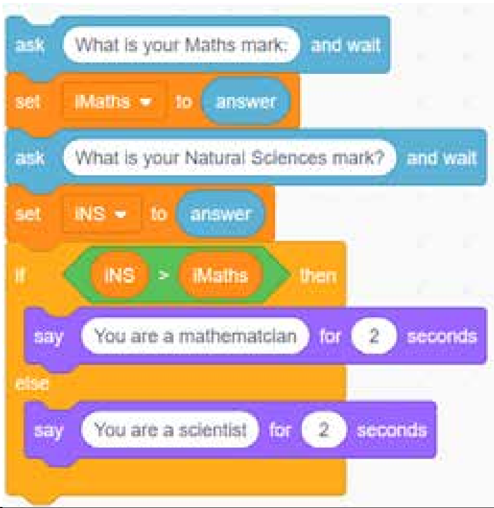
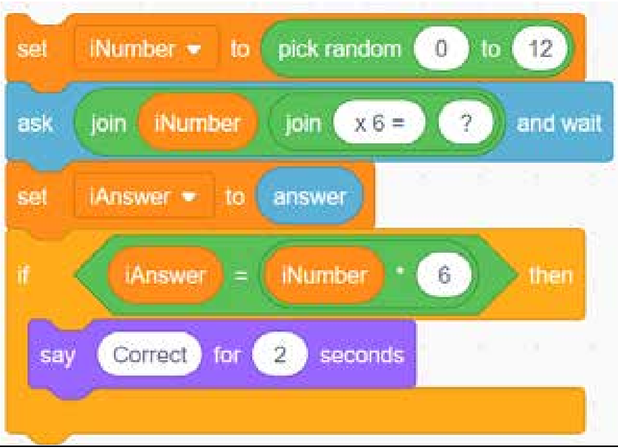
Towards term 4, some elementary problems may be introduced where the learners are expected to deduce that a variable is required. Variables should not be assessed earlier.

Variables should not be over assessed. It should not count more than 5% in any assessment in term 4.

Scratch is not very strict on variable types or variable names. Other programming languages are much stricter with variable types and names.

To ensure good programming practices, names of variables are selected according to certain naming conventions. A naming convention is mainly used so that programmers understand each other's code.

Content (Grade 6 / Term 2)	Notes/Examples
Example activity 3 – Create and change variable each time space bar is pressed Provide learners with the code on the right. Let them run the code and execute the code. Ask them to pay attention to the value of the variable iScore How many times did you press the spacebar? Why is it necessary to set the score to 0 when the program starts (green flag is clicked)? Let learners explain what the code does.  **Example activity 4 – Create and change a variable** Provide learners with the code on the right Let them study the code and predict what the outputt would be Let them run the code and compare the output to their prediction Now, let them explain what the **Change** instruction does.  **Example activity 5 - Consolidation** Teacher provides learners with a task/problem that uses a variable which they need to plan, code, execute, test and debug. **Example activity 6 – Open ended** Leaners use their knowledge, skills and experience to write a program of their choice that uses a variable and some other concepts that they have learned.	Generally, conventions for variables are as follows: • Variable names **describe the data they will contain.** For example: Variable name: "Amount" will contain numbers. • Variables names **start with a single letter prefix describing the data type** of the variable, e.g. iNumber. • Variable names use **CamelCase.** This means the first letter is lowercase and each word thereafter starts with an uppercase, for example, nameSurname. • All variable names must be **unique.** Two variables cannot have the same name. It will confuse the memory of the computer. A computer is not as clever as you think it is. • Variable names **may not contain any spaces**, for example "My Name" is an incorrect variable name, but tMyName would be correct. • Variable names **may not start with numbers** but **may contain numbers**, for example "12Names" is incorrect, but "Names12" would be correct. • Variable names **may not contain any special characters (!,@,#,$,%,^ etc.) except for underscore (_)**, for example "Name&Surname" will be incorrect, while "Name_Surname" will be correct.
C.3 Interpret and execute a given symbolic or written set of commands	
Example activity 1 – Two variables with IF…THEN…ELSE Translate the following algorithm into code: • Ask the user what their Name is • Ask the user their age • If the age is under 19, greet the user, saying Hello [name], your [age] indicates that you are probably still at school. • Else (if age is 19 or more), greet the user saying Hello [name], your [age] indicates that you are probably no longer at school. Run the code, test, and debug. **Example activity 2 – Open ended** Leaners use their knowledge, skills and experience to write a program of their choice that uses a variable and other concepts that they have learned so far.	**Note** In Grade 6, learners are not expected to use more than two variables in one program. The concept of Parsons puzzles is a type of scaffolded program construction tasks where the learner is given a set of code blocks of a single or multiple lines of code, and the task is to piece together a program from these or to fill in missing code from these. It helps learners to develop logical thinking.

Content (Grade 6 / Term 2)	Notes/Examples
Example activities – Parsons puzzle concept **Example 1 – Fill in missing coding instructions using blocks provided** Provide a problem description and a partial program to solve the problem in the scripts area (leave gaps where missing code instructions should be placed). Also provide the missing code blocks randomly placed (not in sequence) in the scripts area. Learners need to figure out where the missing code blocks fit to complete the program and solve the problem. **Example 2 – Complete a program using code blocks provided** Provide a problem description and all the code blocks to solve the problem, randomly placed in the scripts area (not in sequence) Learners then need to **fit the blocks of code together in the correct sequence** to solve the problem and ensure it function correctly.	Parsons programming puzzles are an evidence-based teaching practice that reduces the cognitive load and time spent for learners.
C.4 Debug a given symbolic or written set of instructions	
Example activity Provide learners with the code on the right on paper Provide learners with the following input values to test the code: Learners run the program and enter their marks. The program provides incorrect output. Learners need to figure out what is wrong and correct the code.  	Learners trace code using input values provided. Learners test code and evaluate the output for correctness. Learners correct the code
C.5 Evaluate a given solution towards potential improvement	
Example activity 1 – Improve code Mary wrote a program for her little sister to practise her 6x table. Evaluate the code and improve the code to include the following: If her sister gave the wrong answer, the program must tell her it is wrong Instead of running the program over-and-over again (repeatedly clicking the green flag) if she wants to continue testing her 6x table, improve the program so that her sister could test herself 12 times before it is necessary to click the green flag again. Test your improved code and debug your code after improving it, if necessary.  	Learners improve code to be more efficient. Improvement should include an IF…THEN…ELSE and a REPEAT (12x) loop

Content (Grade 6 / Term 2)	Notes/Examples
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to and R.5 – R.7**

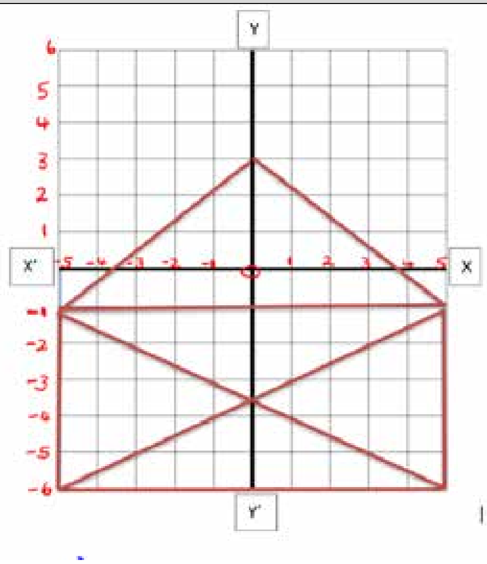
Example activity

You need to draw a house with a cross on the walls as shown on the right: You must follow the following 'rules' when drawing the house:

- You may not lift your hand/pen.
- You may not draw on a line that has already been drawn.

Someone created two algorithms for drawing the house according to the above rules.
Follow each algorithm to see if it complies with the rules above. If you find that the algorithm does not comply with the rules, rewrite it so that it is in line with the rules.
The coordinates are in the format (x,y), e.g. (1, 2) refers to x=1 and y=2 on the grid.

Algorithm 1	Algorithm 2
1. Start at (5, –1)	1. Start at (5, –6)
2. From the above position, draw a diagonal line to (–5, –6)	2. From the above position, draw a diagonal line to (–5, –1)
3. From the position in step 2, draw a straight line to (5, –6)	3. From the position in step 2, draw a straight line to (5, –1)
4. From the position in step 3, draw a diagonal line to (–5, –1)	4. From the position in step 3, draw a diagonal line to (–5, –6)
5. From the position in step 4, draw a diagonal line to (0, 3)	5. From the position in step 4, draw a straight line to (5, –6)
6. From the position in step 5, draw a diagonal line to (5, –1)	6. From the position in step 5, draw a straight line to (5, –1)
7. From the position in step 6, draw a straight line to (5, –6)	7. From the position in step 6, draw a diagonal line to (0, 3)
8. From the position in step 2, draw a straight line to (–5, –1)	8. From the position in step 7, draw a diagonal line to (–5, –1)
9. From the position in step 8, draw a straight line to (5, –1)	9. From the position in step 8, draw a straight line to (–5, –6)

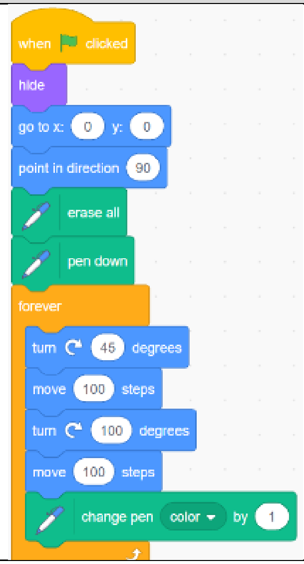


Notes/Examples: After completing the activity, ask learners if there could be more than one starting point to achieve the same outcome.

C.7 Create or complete a pattern to represent a data set

Example activity 1 – Circle pattern

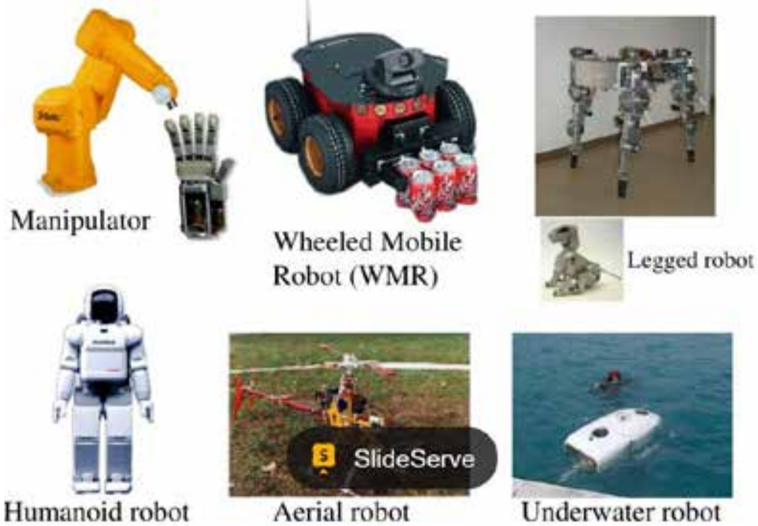
Provide learners with the code on the right
Let them run the code and observe what happens.
Let them explain the code



Example activity 2 Open ended

Leaners use their knowledge, skills and experience to write a program of their choice to create a similar program.

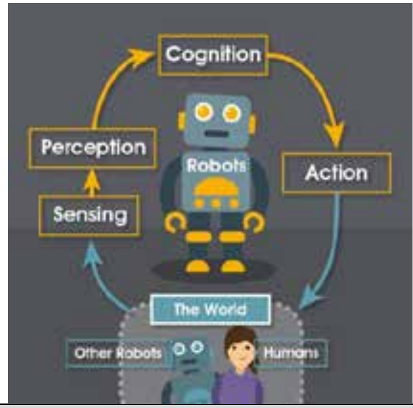
Notes/Examples: Previous experience also plays a big role when we solve problems

Content (Grade 6 / Term 2)	Notes/Examples
Content (Grade 6 / Term 2)	Notes/Examples
Robotics	
R.1 Explain what a robot is in simple terms.	**Link to R.2 and R.3**
R.2 Identify different types of robots.	**R.1 – R.4 can be done together**
Briefly revise what a robot is and summarise different types of robots, e.g. providing pictures and discuss briefly.  **How a robot is controlled** **Input**: The first step in controlling a robot is giving it instructions. Just like we use a remote control to tell a toy car what to do, robots receive instructions too. These instructions can come from a human operator who uses a computer, a tablet, or buttons to give commands to the robot. **Sensors**: Robots have sensors that help them understand what's happening around them. Sensors can detect things like obstacles, sounds, or even light. When the robot receives information from its sensors, it can use that information to make decisions about what actions to take. **Controller**: The controller is like the brain of the robot. It's a special computer that receives the instructions from the human operator and processes the information from the sensors. The controller tells the robot what actions to take based on the instructions and the sensor data. **Actuators**: Actuators are the parts of the robot that make it move or perform tasks. They are like the robot's muscles. The controller sends signals to the actuators, such as motors or hydraulic systems, to tell them how and when to move. For example, if the robot needs to walk forward, the controller will send a signal to the motors in its legs to start moving. **Output**: The output is what the robot does or how it behaves based on the instructions and sensor information. It can be actions like moving, picking up an object with its gripper, or even speaking if the robot has a voice. The robot's output is a result of the controller sending signals to the actuators. **Feedback**: After the robot performs an action, it may use its sensors again to check if the action was successful. This is called feedback. For example, if a robot is programmed to avoid obstacles, it will use its sensors to detect if there's anything in its path. If it senses an obstacle, it will adjust its movements to avoid it. Feedback helps the robot adjust and improve its performance.	**Extend to how a robot is controlled in terms of input, process (role of components such as sensors, controller, actuators) and output** **How a robot is controlled** A robot can be controlled to perform specific tasks. The human operator gives instructions, the sensors provide information about the environment, the controller processes the data and makes decisions, and the actuators carry out the actions. It's like a cycle where the robot takes input, processes it, and produces an output based on that input.
R.3 Outline the different components of a robot	
Robotic interactions with the world **Sensing**: The robot uses its sensors to gather information about its environment. Sensors can include cameras, microphones, touch sensors, and more. Each sensor detects specific aspects of the world, such as detecting light, sound, objects, or even temperature. The robot receives input from these sensors, which provides data about its surroundings.	Learners need to understand: This process of sensing, perception, cognition, and acting enables the robot to interact with and navigate the real world.

<table>
<tr><th>Content (Grade 6 / Term 2)</th><th>Notes/Examples</th></tr>
<tr><td>

Perception: Once the robot has gathered sensory input, it needs to make sense of that information. This is called perception. The robot's perception system processes the sensory data and tries to understand what it means. For example, the robot's camera might capture images of objects, and the perception system analyses those images to recognize and identify different objects in its surroundings.

Cognition: After perceiving the environment, the robot's cognition comes into play. Cognition refers to the robot's ability to think, reason, and make decisions based on the information it has gathered. The robot's cognitive system uses algorithms and programming to analyse and interpret the perceived data. It can use pre-defined rules, machine learning, or artificial intelligence techniques to understand the situation and determine the best course of action.



Taking Action: Once the robot has processed the sensory information and made decisions, it's time to act. The robot uses its actuators, such as motors or grippers, to physically interact with the real world. For example, if the robot has determined that it needs to pick up an object, it will activate its gripper to grasp the object. If it needs to move, it will command its motors to start moving in the desired direction.

Feedback and Iteration: After the robot has acted, it can use its sensors again to gather feedback on the results of its actions. This feedback is essential for the robot to evaluate whether it has achieved its goal or if any adjustments are necessary. Based on the feedback, the robot can modify its future actions, refine its perception or cognition algorithms, and improve its overall performance.

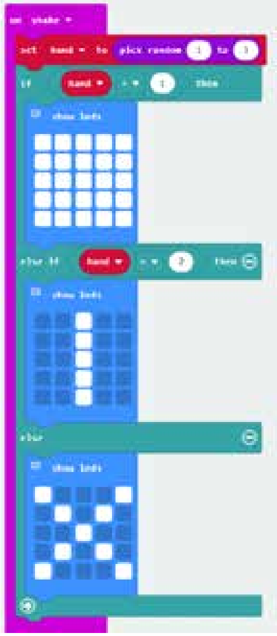
</td><td>

By continuously gathering information, interpreting it, making decisions, and executing actions, the robot can adapt to different situations and carry out tasks in its environment.

This cycle of sensing, perception, cognition, and action helps the robot learn, adapt, and accomplish a wide range of functions and tasks.

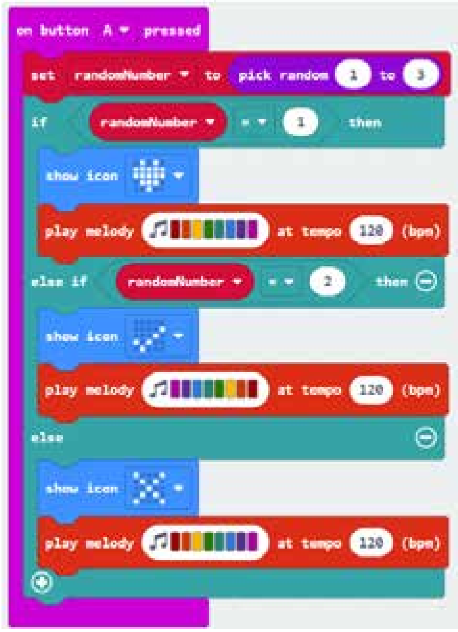
</td></tr>
<tr><th>R.5 Design a simple artefact based on a set of design instructions</th><th></th></tr>
<tr><td>

Example activity 1 – Rock-Paper-Scissors



Example activity 2 – Simulating a dice

Example activity 3 – Musical Faces



</td><td>

Learners are introduced to variables in a very simplistic form. Using the variable relate the concept of the single variable to that of the answer block used in other block-based languages.

Learners should be guided in all activities as to which and when variables are to be declared. The learners should not deduce when variables are to be used in given problems.

Note

It is very important that variables are introduced correctly to avoid future misconceptions.

Learners generally struggle with the concept of variables.

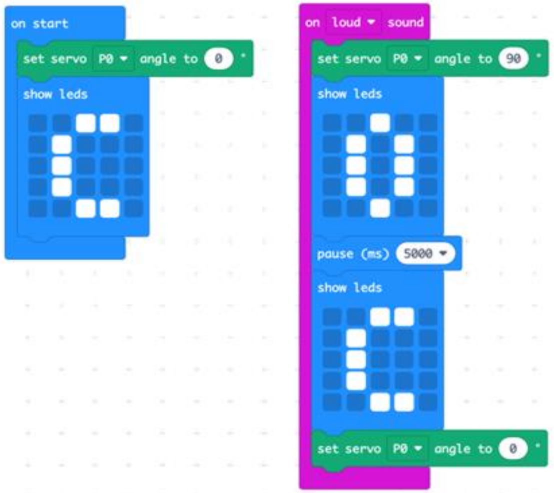
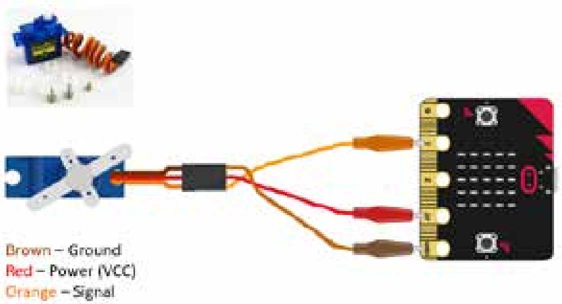
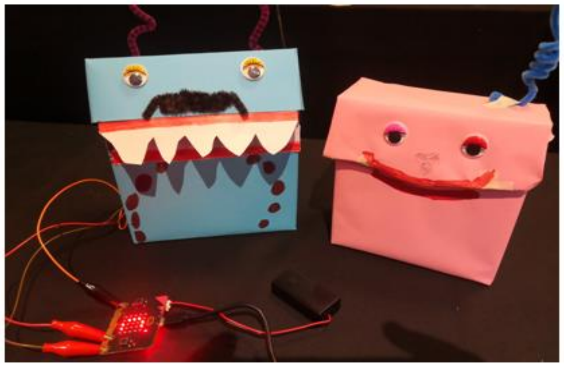
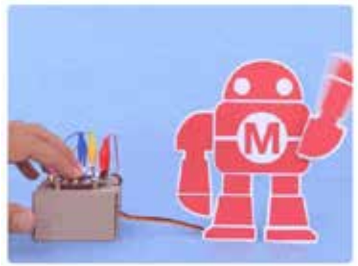
Learners generally struggle with conceptual understanding of introductory programming concepts such as variables, expressions, and loops (Grover et al, 2019). They suggest that conceptual exploration support preliminary engagement with and learning of foundational and often hard-to-grasp programming concepts for novices (Grade 6 – 8 learners).

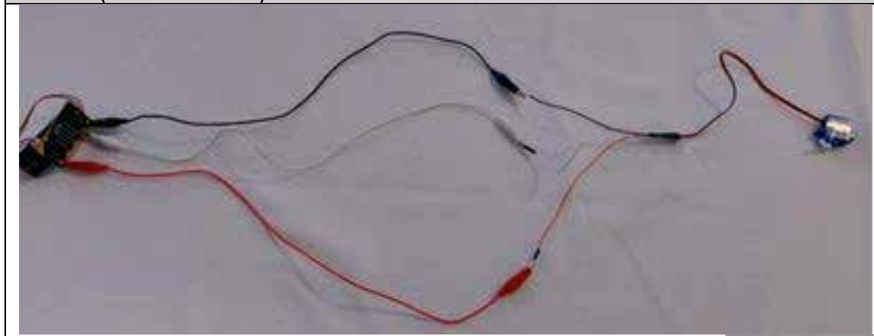
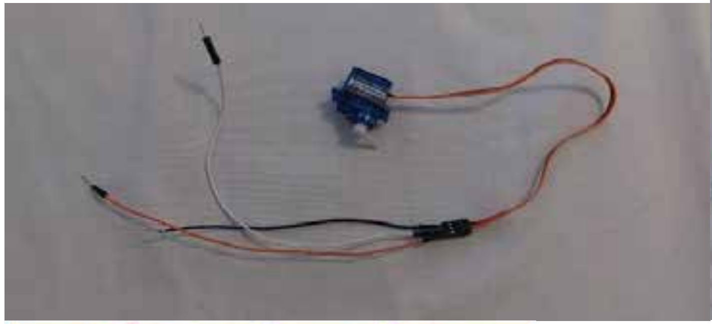
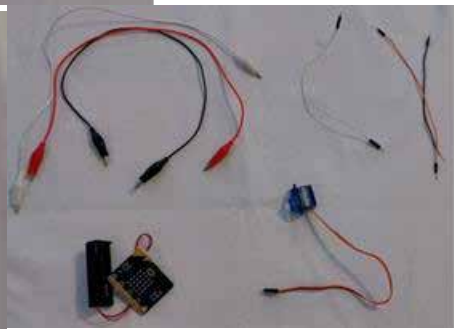
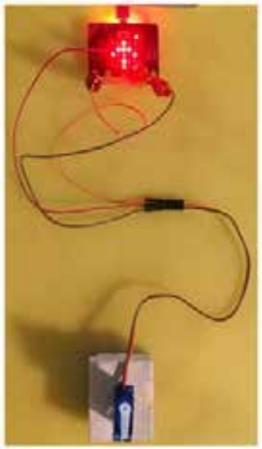
Variables in coding are similar but not the same as in Maths. In coding, variables should be introduced as objects that can be manipulated rather than just a placeholder for an unknown value.

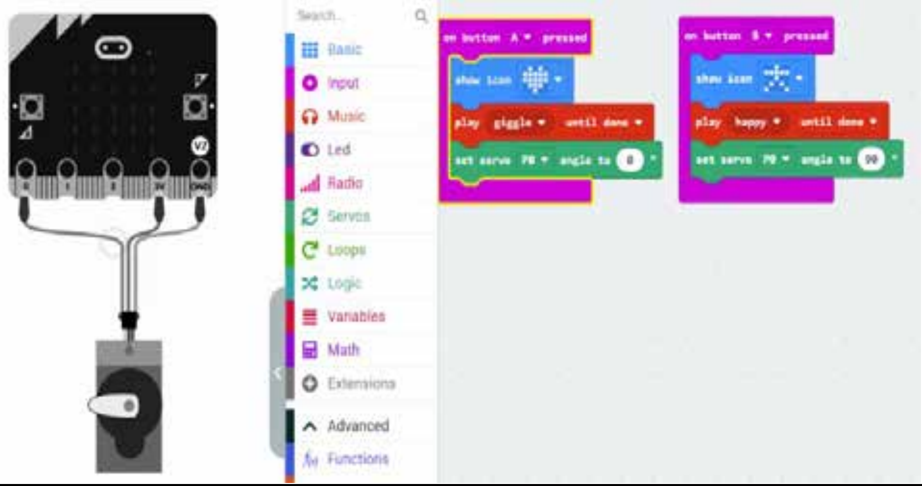
Leaners need to understand that variables can be set or changed by code and that some variables affect other variables within the code.

Variables have a value and a name, but also a type. Certain operations in coding can only be performed on certain types.

</td></tr>
</table>

Content (Grade 6 / Term 2)	Notes/Examples
R.6 Mimic the operations of a robot	**Link to R.5 and C.1 – C.7**
Example activity - Musical die Write code to do the following: Shake a microcontroller to display a random number between 1 and 6 that can be used as a die. Choose a sound for each number and display the number and make a sound that is associated with that number.	**Link to R.5 and R.7**
R.7 Create, test and execute a set of robotic instructions	**R. 6 and R.7 are done together with R.5 (once enabling activities in R.5 are completed) to complete the project**
Project – Exploring servo motors Use computational thinking, design thinking and the engineering design process plan, develop, execute and test a set of instructions to program the servo motor to control the opening and closing of the animal's mouth.  Servo inside an empty tea box. This tea box was used to create the blue box monster with a moustache.   	Learners need to demonstrate • how to use a microcontroller (e.g. micro:bit) to control a servo motor • Knowledge of the working principles of servo motors, • how to connect them to the microcontroller pins, and • how to control their movement using code Other examples of projects to be considered: ***Example B - Waving hand*** Microbit Waving hand (Servo) https://www.wonkitz.com/classroom-activities/motion/microbit-how-to-use-a-servo-to-make-a-waving-hand/  https://www.wonkitz.com/classroom-activities/motion/microbit-how-to-use-a-servo-to-make-a-waving-hand/ https://learn.adafruit.com/makey-paper-craft/overview  Straw Bot https://classroom.strawbees.com/resource/make-a-facebot-microbit#materials

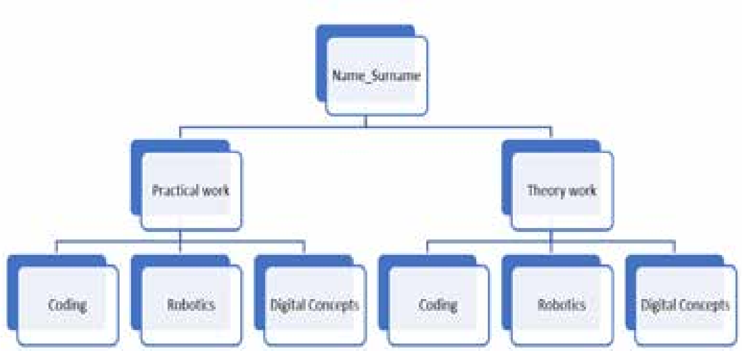
Content (Grade 6 / Term 2)	Notes/Examples
    	 **Highlight the concept of an end effector.** An end effector is an important part of a robot that helps it interact with the world around it. Think of it as the "hand" or "tool" of the robot. Just like we use our hands to pick up objects or perform tasks, the end effector is what the robot uses to do its work. The end effector can come in different shapes and sizes depending on the robot's purpose. It can have things like grippers, claws, or specialised tools attached to it. For example, a robot in a factory might have a clamp-like end effector to pick up and move objects. Another robot in a laboratory might have a small arm with a precise tool for conducting delicate experiments. The end effector is usually located at the end of the robot's arm or manipulator. It can be controlled by the robot's computer or by a human operator. The robot can use its end effector to perform tasks like picking up objects, assembling parts, painting, or even playing games. In simpler terms, the end effector is like the robot's hand that allows it to interact with its environment and perform different tasks. It's an essential part of the robot that helps it be useful and perform its designated job.

Content (Grade 6 / Term 2)	Notes/Examples
	
Digital Concepts	
D.2 Recognise that he or she is living as citizens in a digital world.	
Reinforce and extend from the previous grades and terms using different examples and activities. • Understand ethical issues and dangers associated with the use of information technology, including privacy, security, copyright, false information and inappropriate content. **<u>Example activity:</u> Dangers associated with IT.** Divide the learners into small teams. Provide each team with a set of printed scenario cards describing various IT dangers. Each scenario should outline a potentially risky situation that users may encounter while using information technology. Instruct the teams to read and discuss the scenarios. Ask them to identify the potential dangers and suggest safe actions to handle the situations.	Digital citizenship can be defined as the quality of habits, actions and consumption patterns that impact the ecology of digital content and communities
D.4 Identify the common uses of ICT in the real world	**Link to D.5, D, 7 and C.7, R.3 – R.7**
D.5 Differentiate between the components of an ICT system	**Link to D.4, D,7 and C.7, R.3 – R.7**

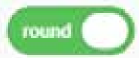
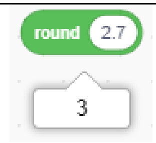
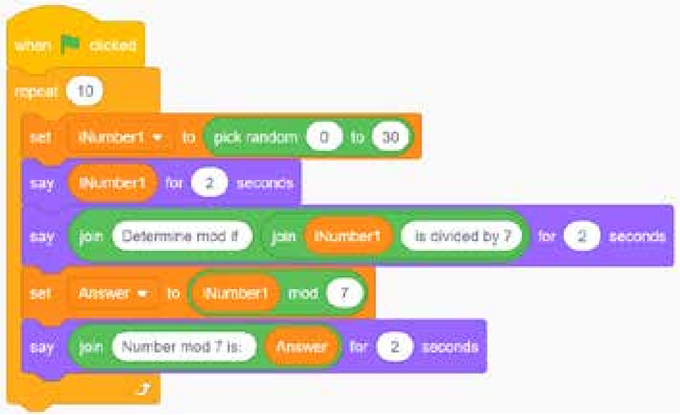
Reinforce and extend from the previous grades and terms using different examples and activities.
A basic ICT system consists of the following basic components:
• Hardware (computing devices (computer and microcontroller), USB cable (allowing communication to take place)
• Software (microcontroller app on the computer to code instructions for the microcontroller)
• Data (input and output via e.g., input devices/buttons, sensors)
• People (the users of the system)
 	
<u>Example activity:</u> Connecting a Microcontroller to a Computer (communication between two computing devices – elementary ICT system)
Explain to learners that one needs to connect the microcontroller to the computer to allow the code to be executed on the microcontroller. The connection allows us to write instructions and send the instructions and data to the microcontroller.
Components Required for Connection
Show the physical microcontroller board to the learners and explain its different parts, such as input/output pins, power connector, and USB port.
Discuss the USB cable and its role in establishing a connection between the microcontroller and the computer.
Explain that the USB cable is used for both supply power and data communication between the microcontroller and the computer.
Connecting the Microcontroller
Demonstrate the steps of connecting the microcontroller to the computer using a USB cable.
	D.4 and D.5 is done together
	Learners need to
	Know that and ICT system is:
	• Diverse set of technology tools and resources used to communicate, create, disseminate, store and manage information.
	• An ICT System is focused on managing data and information, e.g., Point of Sale System (POS)
	• Name common uses of ICT systems in the real world
	• Know that an ICT system consists of four basic components.
	  • Hardware
	  • Software
	  • Data
	  • People
	
	Learners need to:
	Know that connecting the microcontroller to the computer allows communication between two computing devices (instructions and

Content (Grade 6 / Term 2)	Notes/Examples
Guide the learners through the process of connecting the LED circuit to the microcontroller and ensuring the USB cable is properly connected to the computer. Software and Programming Explain that to communicate with the microcontroller, we need software called an Integrated Development Environment (IDE). Demonstrate a simple program being downloaded to the microcontroller and that turns the LED on and off using the microcontroller. Hands-on Activity Allow the learners to experiment with the microcontroller by changing the program to make the LED blink at different intervals. Encourage creativity and problem-solving, letting them explore various programming options.	data sent from the computer to the microcontroller), resulting in an elementary network. Input processing and output of an ICT system (link to C.7)
D.6 Explain how the adaptation of technology impacted the world we work and live in	
Example activity - False information (extend from Term 1) Divide learners into pairs and provide learners with a worksheet with questions to answer. Learners now watch the following videos: https://youtu.be/BIv9054dBBI and https://youtu.be/KX8-BOc7Z0c Learners answer the questions while watching the videos. Teacher discusses the questions. Possible discussion questions • What is fake news? • Types of fake news • Why do people create fake news? • How do we explain the difference between fake news and facts? • What roles do social media play in spreading fake news? • How can fake news be identified? • Who benefits from fake news? • Does fake news have any victims? • What are the consequences of fake news? • How can fake news be prevented? • Fake news – humans or bots or both? • What are social media platforms doing to prevent the spread of fake news? • What is the right thing to do when you spot fake news?	Learners need to • understand what fake/false news is • Know types of fake news • How fake/false news is spread • provide examples of fake news • Identify fake news • How to respond to fake news • how fake news/false information could impact our lives
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.4 and D.5**
Reinforce and extend from the previous grades and terms using different examples and activities. Discuss input, processing and output of a basic ICT system	Done with D.4 and D.5
D.8 Interpret a pattern to represent or communicate a message or image	**Link to C.6 and C.7**
Reinforce and extend from the previous grades and terms using different examples and activities. Create patterns using the coding app and microcontroller app	
D.9 Create a pattern to represent or communicate a message or image	**Link to C.6 and C.7**
Reinforce and extend from the previous grades and terms using different examples and activities. **<u>Example activity:</u> Transfer a pattern on paper to a program.** Continue with activity in D.8. in Term 1. Learners created a smiley face on paper. Let them convert this to Scratch by: • Draw the smiley face using familiar tools such as Paint. – smileys should have different costumes • Import the sprite(s) into the block-based coding app • Add code blocks to switch between Sprites (costumes can be created for different smiley faces). • Write a story using the smileys using Add a Forever loop and delays.	
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.2 – C.5, and R.5 – R.7**
Reinforce and extend from the previous grades and terms using different examples and activities.	Coding and Robotics activities done on the computer.

Content (Grade 6 / Term 2)	Notes/Examples
Basic folder and file management and naming conventions **Example activity** Provide learners with diagram of a folder structure on a worksheet, e.g. Learners: • create the folder structure on their computing device • write down paths to specific folders • create files which they save in a specific folder • Let them add/delete/move folders Explain what happens when folders (and files) are deleted (concept of recycle bin) **Example activity 2** Use Paint to create sprites and backgrounds to import to block-based coding app Save the files in the correct folder using descriptive names, etc.	



3.3.3 Term 3

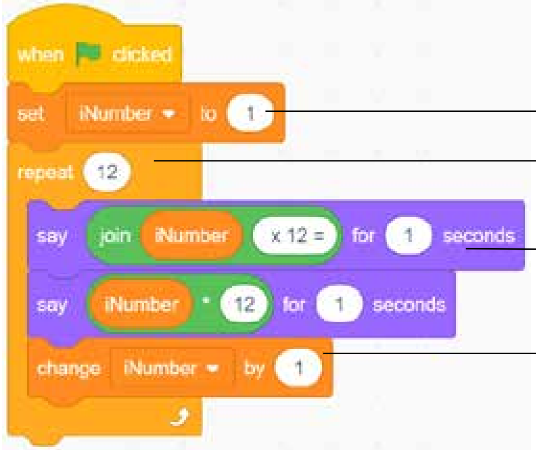
<table>
<tr><th>Content (Grade 6 / Term 3)</th><th>Notes/Examples</th></tr>
<tr><td>C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.</td><td>Link to C.2 – C.7 and R.5 – R.7</td></tr>
<tr><td>Example activity
Watch the following video and complete the following activity:https://youtu.be/Il7tuwpifIM
In a group of 2-3, write instructions for a 1-minute dance sequence
Swap your dance instructions with another group and see if you can correctly carry out each other's moves.
If your instructions are not specific enough, revise them and test them again.</td><td>Use computational thinking to develop instructions</td></tr>
<tr><td>C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.</td><td></td></tr>
<tr><td>Operations in programming
You have learned about the following calculations: plus, minus multiply and divide (+ - * /):
One can also do the following: integer division (mod) and round
Note that the symbol for multiplication (*) and division (/) differ from what we do in maths
<table>
<tr><th>Integer division</th><th colspan="2">Round</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><td></td><td></td><td></td></tr>
</table>
Example activity 1 – Calculations
Vusi wrote the following program for his litte brother to practise integer division.

In pairs, run the code and explain the code line-by-line (instruction block-by instruction block) and write down what the result of each calculation would be.

Example activity 3 – Use and change a variable
Divide learners into pairs. On a worksheet, provide each pair with the following code:
Now, learners study the code provided below:</td>
<td>Remind learners that when doing calculations, the order of preference (BODMAS) is the same as with mathematics.
Explain to learners how integer division (mod) works.

Note:
Many learners tend to focus on very small parts of the code and lose sight of the "big picture".
They are also prone to focus on superficial aspects of the task/problem that are not functionally central to the solution (Lister & Teague, 2014)

Note: Provide learner with activities enabling them to
read code and explain what it does or
work through (trace) / act out code (physically or simulated) to determine the output or the correctness or
provide missing code instructions (code instructions are provided with some instructions or code elements missing) that learners need to complete or
translate verbal/written instructions (algorithm) to code (e.g. write block-based code for a list of symbolic (e.g. arrows)/written instructions))
add some functionality/instructions to an existing program.
rewrite a set of coding instructions to be more efficient, e.g. using a loop construct for code that is repeated or
choose the correct solution from 2-3 options or
compare different solutions to evaluate efficiency or
debug an algorithm or block-based program (find the bug, describe the bug and correct it)
develop a solution/algorithm (code instructions) based on a given problem or for an open-ended problem through planning, implementing, testing and debugging.</td></tr>
</table>

Content (Grade 6 / Term 3)	Notes/Examples



Set variable to 1 (must start at 1 for **1** x 12)

The instructions in the loop will be repeated exactly **12** times to mimic the 12x table

The first time the loop executes it will calculate 1 x 12

Increase variable by 1 to get the next number: for **2** x 12, then **3** x 12….**12** x 12 (The code now changes the value of the variable (overwrites the previous value)

Learners first describe the code line-by-line (what each line (block) of code does)
Then learners explain what the program does.
Leaners now need to understand that, if we want to display the 12-times table, we could use a loop that executes exactly 12 times.
We need to use a variable that starts at 1, and increases by one, each time the loop executes to display all the answers from 1 x 12 up to 12 x 12.
Let learners now write a program for the 7 times table using the above example.

1 x 12 = 12
2 x 12 = 24
:
:
11 x 12 = 121
12 x 12 - 144

You want to interleave practise in problems.
"It is important that problem types must differ, for example, you want to randomly have a problem of one type and then solve a problem of another type and then a problem of another type. And in doing that, it feels difficult, and it doesn't feel fluent. And the signals to your brain are, I'm not getting this. I'm not doing very well. But in fact, that effort to try to figure out what kinds of approaches do I need for each problem as I encounter a different kind of problem, that's producing learning. That's producing robust skills that stick with you."
Dr Mark A. McDaniel, Harvard University

Note:
Coding is not just about writing code; it's about thinking logically and algorithmically and developing a problem-solving mindset that transcends specific programming languages and technologies

C.3 Interpret and execute a given symbolic or written set of commands

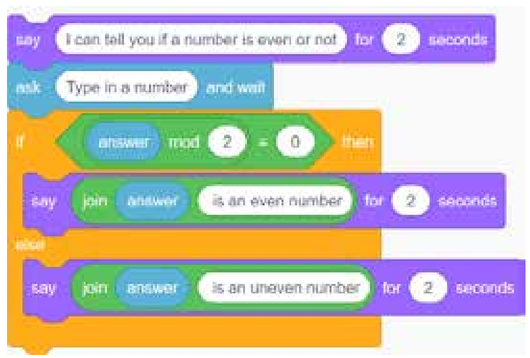
Example activity 1
Code the following high-level algorithm:

- A user must type in a number
- The program must determine if the number entered is even or odd
- The program must display the number that the user typed in and tell the user whether the number is even or odd

Example activity 2
Provide learners with the code on the right.
Learners study the code and explain what it does.

Example activity 2 – Open-ended
Leaners use their knowledge, skills and experience to write a program of their choice that does something similar.

Possible answer for activity 1



Content (Grade 6 / Term 3)	Notes/Examples
C.4 Debug a given symbolic or written set of instructions	**Link to C.2, C.3**

Example activity – Debug a program using a variable

Divide learners into pairs. Each pair has a driver and a navigator

The driver loads the program, runs it a few times using different values for input (name and age - use different names and ages each time the program is run) The navigator must provide the output (write down the input and the corresponding output) each time that the program is run.

Do they notice the mistake in the output? The navigator points out the mistake in the output (can draw a red circle on the output that was written down)

Why is there a mistake in the output? (Study the programming code.)

The pair changes roles (swop navigator and driver) and corrects the programming code, then tests the program again using different data sets (different names and ages) to check if the problem is solved/error has been corrected.

Each pair now explains what the problem was and how it was corrected.

Notes/Examples: It is important that learners discover the error themselves and correct it themselves.
Teacher can provide the datasets for testing if necessary

C.5 Evaluate a given solution towards potential improvement

Provide learners with code which need improvement and ask them to improve the code, e.g. repetitive steps where a loop can be used or multiple if...then statements which can be replaced by an If…then…else, etc.

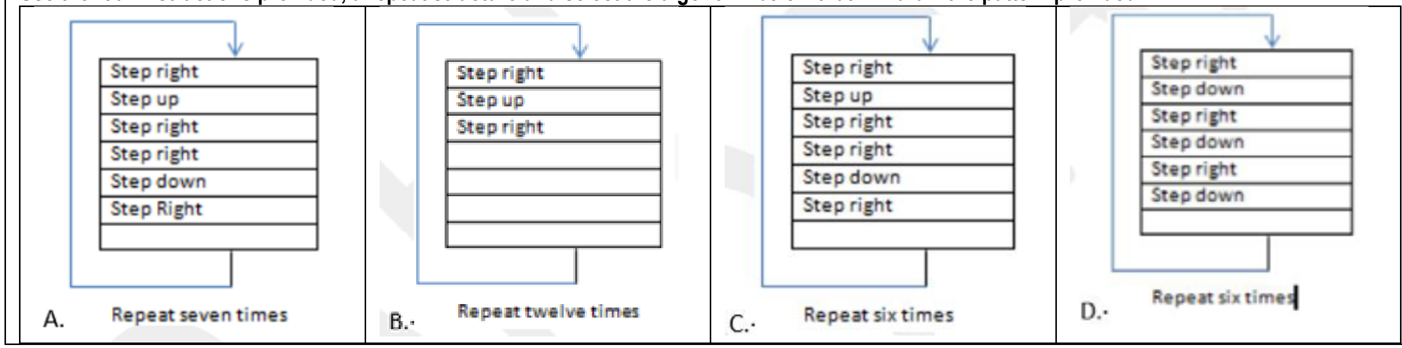
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.

Example activity 1

You are provided with four instructions and a pattern:

Walk 1 step to the right
Walk 1 step to the left
Walk 1 step up
Walk 1 step down

Use the four instructions provided, a repeat structure and select the algorithm below that will draw the pattern provided



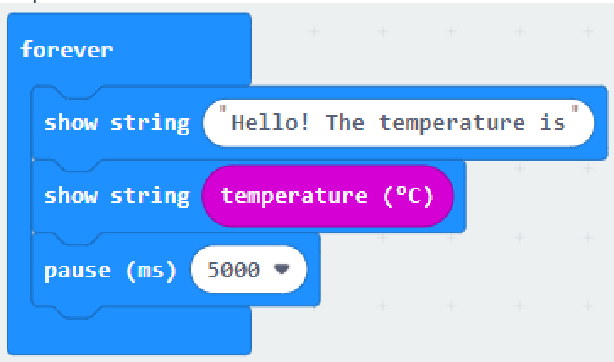
Notes/Examples: Pattern recognition is part of computational thinking and is used to identify patterns in coding problems and/or data by identifying similarities or differences that can help to solve the problem or refine the algorithm.

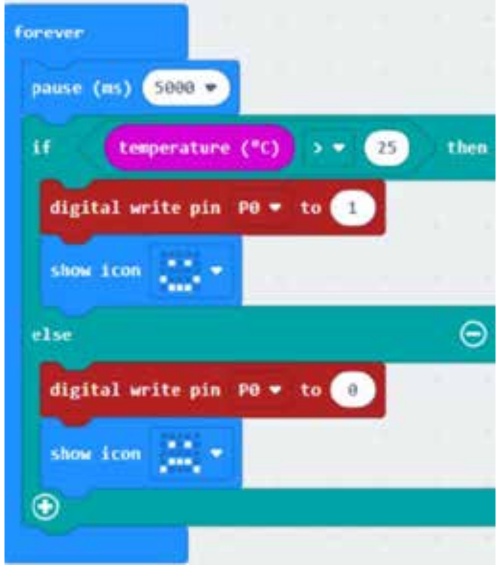
<table>
<tr><th>Content (Grade 6 / Term 3)</th><th>Notes/Examples</th></tr>
<tr><td>Robotics</td><td></td></tr>
<tr><td>R.4 Present an understanding of how robots affect the world</td><td>Link to R.5 – R.7</td></tr>
<tr><td>Example activity
Discuss with learners benefits and risks of robots
Benefits:
• Making work easier and faster: Robots can do difficult, dangerous, or repetitive tasks, freeing up humans to focus on more creative or complex work.
• Saving time: Robots are speedy and efficient, completing tasks faster than humans.
• Helping people: Robots can assist individuals with disabilities, perform surgeries, and provide companionship.
• Exploring new frontiers: Robots can go to places humans can't reach, like Mars or deep oceans, to gather information and expand our knowledge.
• Entertainment and fun: Robots can be toys or appear in movies and shows, providing enjoyment and excitement.
Risks:
• Job displacement: As robots can perform tasks previously done by humans, some jobs may no longer be needed, leading to unemployment for some workers.
• Dependence on technology: If we rely too much on robots, we may become dependent on them and lose certain skills or abilities.
• Privacy and security: Robots equipped with cameras or sensors can raise concerns about privacy invasion or potential security risks if they are hacked or misused.
• Ethical considerations: As robots become more advanced, questions arise about their use in areas like warfare or decision-making where human judgment is important.
• It's essential to strike a balance, using robots in ways that benefit society while considering potential risks and addressing any ethical concerns.
Ethical considerations towards the use and implementation of robots
• Keeping people safe: Robots should be designed to make sure they don't hurt people and to follow safety rules.
• Making good decisions: Some robots can make decisions by themselves. We need to think about how much freedom they should have and who is responsible if something goes wrong.
• Being honest: Robots should be programmed in a way that we can understand why they do things. They shouldn't keep secrets or do things without telling us.
• Protecting privacy: Robots can collect personal information, like our names or where we live. We need to make sure that this information is kept safe and used properly.
• Fairness for everyone: Robots should be available to everyone and not just some people. We need to make sure that everyone has a chance to use them.
• Thinking about jobs: Sometimes robots can take over jobs that people used to do. We need to find ways to help people find new jobs and make sure everyone has a fair chance.
• Being good for the environment: We need to think about how robots are made and used. We should make sure they don't harm the environment and that they are made in a way that is good for the Earth.</td><td>Focus on benefits and risks and ethical considerations.
Robots have both benefits and risks associated with their use.
Also, extend to focus on ethical considerations when using robots

Learners need to list and briefly describe
• Benefits and risks of using robots
• Ethical considerations of using robots</td></tr>
<tr><td>R.5 Design a simple artefact based on a set of design instructions</td><td>Link to R.6 – R.7 and C.1 – C.7 and D.6 and D.7</td></tr>
<tr><td>Example activity 1 – Temperature display   Example activity 2 – LED temperature alert</td><td>u</td></tr>
</table>

Content (Grade 6 / Term 3)	Notes/Examples

Use the built-in temperature sensor and display the current temperature on a LED matrix.



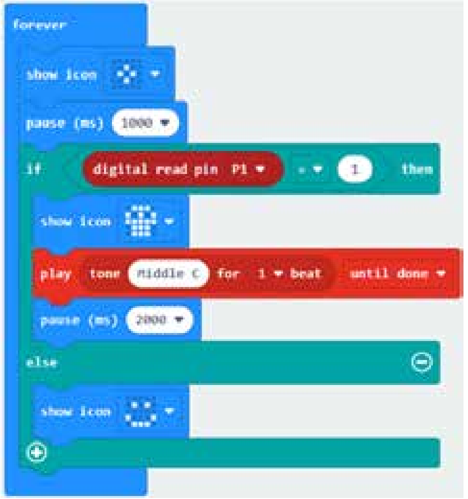


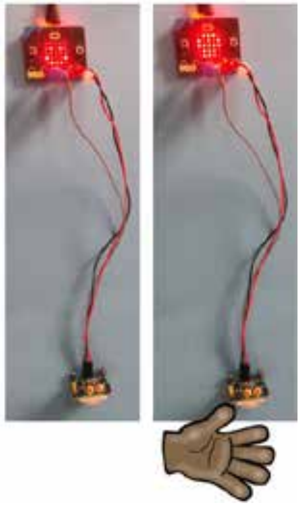
Example activity 3 – PIR Alarm

It is important to double check the PIN configuration on all electronic components before it is used in projects. Ensure that ground, is connected to ground and VCC to voltage and the signal channel to an appropriate pin.

An SR505 is also a good alternative and does not require any fine tuning.

Content (Grade 6 / Term 3)	Notes/Examples

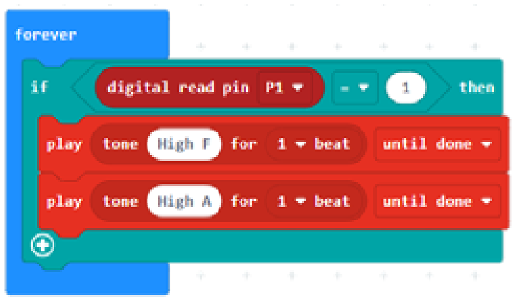


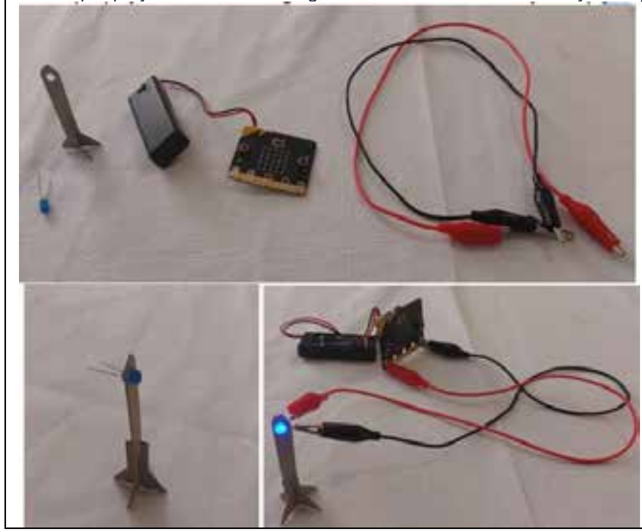
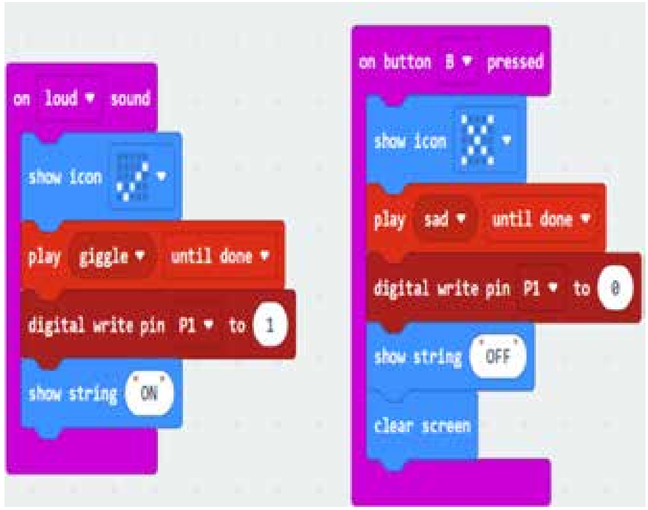


(Also see basic cardboard alarm)

https://microbit.org/projects/make-it-code-it/pressure-switch-alarm/

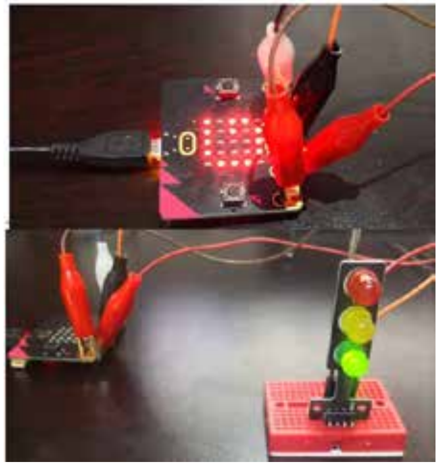
Alternate version: Using large ice cream sticks

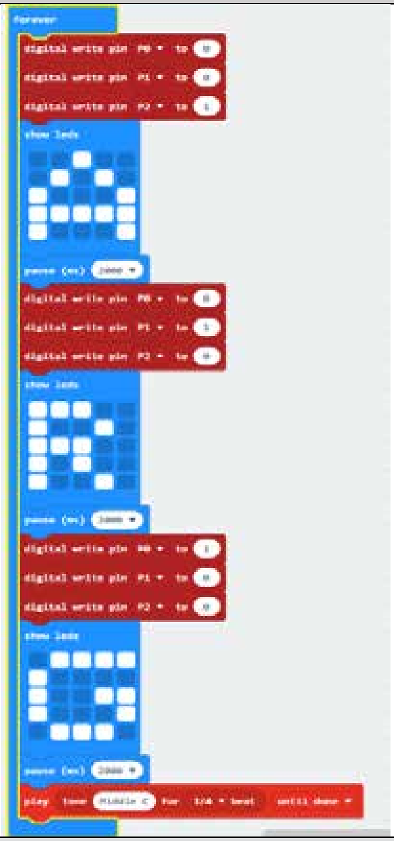


Content (Grade 6 / Term 3)	Notes/Examples
R.6 Mimic the operations of a robot	**Link to R.5 and C.1 – C.7**
Example activity 1 - Microcontroller pet mood detector Write code to do the following: Shake the microcontroller to generate a random number that will indicate the mood or state of your pet. Display the pet's mood or state. Number 0: Pet is hungry Number 1: Pet is happy. Number 2: Pet is tired. Number 3: Pet wants to play. Number 4: Pet is scared. Number 5: Pet loves you.	
R.7 Create, test and execute a set of robotic instructions	R. 6 and R.7 are done together with R.5 (once enabling activities in R.5 are completed) to complete the project
Project – Sound-Controlled Light Pole Develop a project to turn on the light and set off the alarm when you clap your hands.  	

Content (Grade 6 / Term 3)	Notes/Examples

Alternative project (3 LED Trafic Light)





This project shows, A, R, G depending on the state of the lights.

GREEN = P0

RED = P1

Digital Concepts

D.2 Recognise that he or she is living as citizens in a digital world.

Link to D.6

Example activity: Understand your Digital Footprint

Provide learners with a worksheet with the following introduction and questions

Everyone who uses the Internet has a digital footprint. It is wise to consider what trail of data you are leaving behind in the online world. Understanding your digital footprint may prevent you from sending a hurtful email, since the message might remain online forever. It may also guide you to be more sensitive in what you publish on social media websites. While you can often delete content from social media sites, once digital data has been shared online, there is no guarantee you will ever be able to remove it from the Internet.

Now, watch the following videos: https://youtu.be/RHNLGaVRxyI and https://youtu.be/dmQGq_FNBpE

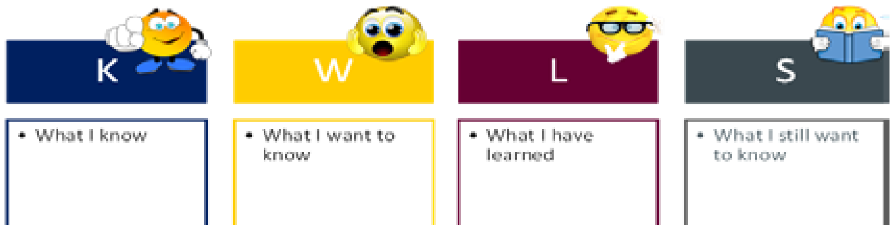
Answer the following questions about how your digital footprint could impact your live?

1. What is a digital footprint?
2. How is a digital footprint created?
3. How can you find your digital footprint?
4. What are examples of a digital footprint?

Digital Passport™ by Common Sense Education

Provide learners with guidelines on how to manage

- Cyberbullying
- Passwords/pins.
- Sharing of personal information.
- Digital footprints

Content (Grade 6 / Term 3)	Notes/Examples
5. How is a digital footprint used? 6. Why is it important to understand your digital footprint? 7. What are the consequences of a digital footprint? 8. How can you manage your digital footprint? 9. What advice can you give to people about digital footprints? 10. What are the online activities leading to a digital footprint?	
D.3 Demonstrate an understanding of the concept of a computing device.	**Link with C.3**
Reinforce and extend from the previous grades and terms using different examples and activities. **Example activity 1:** Microcontrollers as Computing Devices • Learners program a microcontroller, which involves giving it a set of instructions to perform specific tasks. **Example activity 2** • Provide learners with real live examples, e.g., pictures of various everyday devices that contain microcontrollers. • Explain how each device uses a microcontroller and what tasks it performs (e.g., a microwave oven uses a microcontroller to set cooking time and temperature). • Discuss the advantages of using microcontrollers in these devices, such as automation and precise control.	Learners need to • Know what a microcontroller is • Know what input, processing and output is in the context of a microcontroller • Know the CPU is the brain of the microcontroller, responsible for processing instructions • Know that I/O ports (e.g., sensors, buttons) allow communication with external devices • list example of every device that use microcontrollers (e.g., as microwave ovens, remote controls, washing machines, and traffic lights)
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to R.1 – R.4**
Reinforce and extend from the previous grades and terms using different examples and activities. **Example activity: Understanding how Technology has transformed our lives** Give learners the following as a printout. Hand out a KWLS chart or let learners draw one in their learner books.  Technology has transformed our lives in many ways, both positive and negative. On the one hand, technology has enabled us to communicate, learn, work, and entertain ourselves more easily and efficiently. For example, we can use smartphones, laptops, and tablets to access the internet, social media, and online platforms. We can use online courses, e-books, and podcasts to learn new skills and knowledge. We can use email, video conferencing, and cloud computing to work remotely and collaboratively. We can use streaming services, gaming consoles, and virtual reality to enjoy various forms of entertainment. On the other hand, technology also has some drawbacks and challenges. Technology can be addictive, distracting, and isolating. For instance, we can spend too much time on our devices, neglecting our other responsibilities and interests. We can lose our focus and concentration, as we are constantly bombarded by notifications and messages. We can feel lonely and depressed, as we lack face-to-face interactions and meaningful connections. Technology can also pose risks to our privacy, security, and environment. For example, we can expose our personal data and information to hackers, scammers, and advertisers. We can become victims of cyberattacks, identity theft, and online harassment. We can contribute to the pollution and depletion of natural resources, as we consume more energy and generate more electronic waste. Technology can create social inequalities, ethical dilemmas, and cultural conflicts. For instance, we can face digital divide, discrimination, and exclusion based on our access to and use of technology. We can encounter moral and legal issues related to artificial intelligence, biotechnology, and genetic engineering. We can experience cultural clashes and misunderstandings due to the diversity and complexity of the online world. Technology can also affect our mental and physical well-being, as well as our interpersonal relationships and social skills. For example, we can suffer from stress, anxiety, insomnia, and eye strain due to the excessive use of technology. We can develop poor posture, obesity, and chronic diseases due to the lack of physical activity and healthy habits. We can have conflicts, arguments, and breakups due to miscommunication and misunderstanding caused by technology. Let learners use the KWLS chart to identify what they know, what they want to know, what they have learned and what they still need to know.	

<table>
<tr><th>Content (Grade 6 / Term 3)</th><th>Notes/Examples</th></tr>
<tr><td>

Reinforce and extend from the previous grades and terms using different examples and activities.

<u>Example activity:</u> Using technology to enhance learning.

Ask each learner to take some time to think about their interactions with technology in the classroom and during remote learning experiences. Prompt them to consider questions such as:

- How has technology been used in their classrooms or learning environments?
- What digital tools or resources have they found helpful?
- Have they encountered any challenges or drawbacks related to technology in their education?
- How has technology impacted their engagement, understanding, and overall learning experience?

</td><td>

- Interaction with others
- Access to information
- Entertainment (Movie/Audio streams, music instruments, games)

</td></tr>
<tr><th>D.7 Present a basic understanding of the concept of input processing and output.</th><th>Link to R.5 – R.7</th></tr>
<tr><td>

Reinforce and extend from the previous grades and terms using different examples and activities.

<u>Example activity:</u> Understanding of the concept of input processing and output using a microcontroller

Scenario: Smart Plant Care System

The smart system has sensors for soil moisture, light intensity, and temperature. The desired levels are set for soil moisture (60%), light intensity (800 lux), and temperature (20°C to 25°C).

The system collects data from the sensors and evaluates the plant's conditions.

Based on the data, the system waters the plants when soil moisture is low, adjusts lighting when intensity drops, and cools the environment if the temperature exceeds the preferred range. It ensures the plants receive optimal care for healthy growth.

Create an IPO table to decompose the plant care system scenario, e.g.

<table>
<tr><th>Input</th><th>Processing</th><th>Output</th></tr>
<tr><td>Soil moisture sensor</td><td>Microprocessor evaluates soil moisture level</td><td>Activates watering system if soil moisture is below the desired level (60%)</td></tr>
<tr><td>Light intensity sensor</td><td>Microprocessor analyses light intensity</td><td>Adjusts lights to maintain the preferred light intensity (around 800 lux)</td></tr>
<tr><td>Temperature sensor</td><td>Microprocessor checks the temperature</td><td>Activates cooling system if the temperature exceeds the preferred range (20°C to 25°C)</td></tr>
</table>

The IPO table summarizes the inputs, processing, and corresponding outputs of the Smart Plant Care System. Inputs from various sensors provide data on soil moisture, light intensity, temperature, occupancy patterns, and potentially other environmental factors. The microprocessor processes this data to determine the appropriate actions needed to care for the plants optimally. Outputs include activating the watering system, adjusting the artificial grow lights, activating the cooling system, and customizing plant care actions based on occupancy or specific environmental conditions. This smart system ensures the plants receive optimal care for their growth and health.

</td><td>

Learners need to

- know that computing devices uses input, processing and output
- plan input, processing an output when programming a microcontroller

</td></tr>
<tr><th>D.8 Interpret a pattern to represent or communicate a message or image</th><th>Link to D.9, C.1 – C.7, R.5 – R.7</th></tr>
<tr><td>

Reinforce and extend from the previous grades and terms using different examples and activities.

<u>Example activity:</u> Use a logic game to use patterns to convey a message

The beavers are playing a logic game that involves drinking orange juice. John can drink from a bottle when both of the following rules are met:

A) there is a bottle with less juice immediately to the left of this bottle, and

B) there is a bottle with more juice immediately to the right of this bottle.

Which bottles can John drink from:

</td><td>

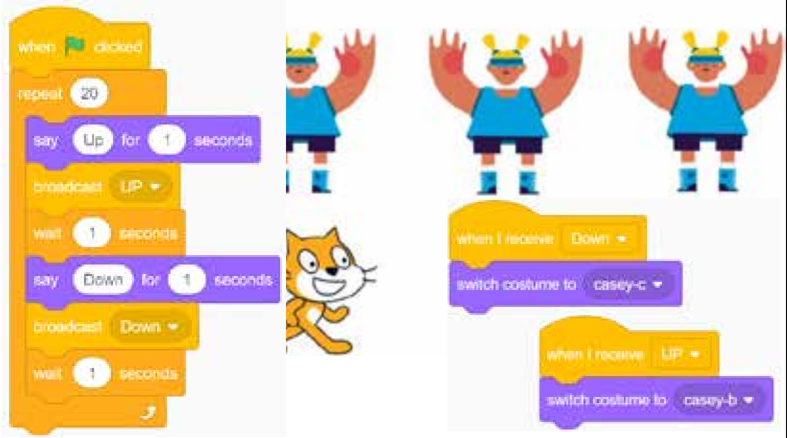
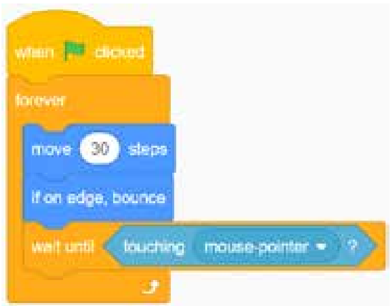
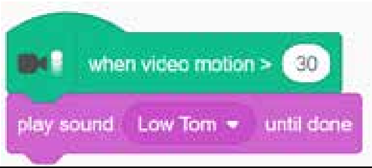
Learners need interpret a pattern that represents or communicates a message or an image using

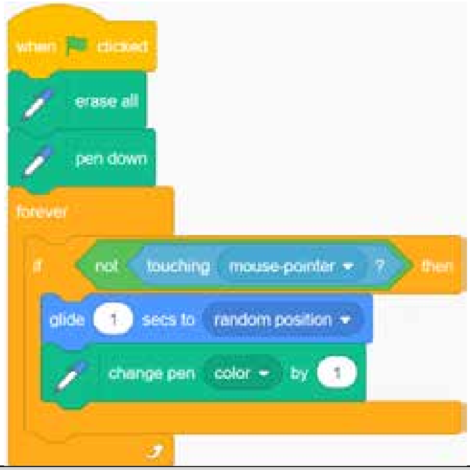
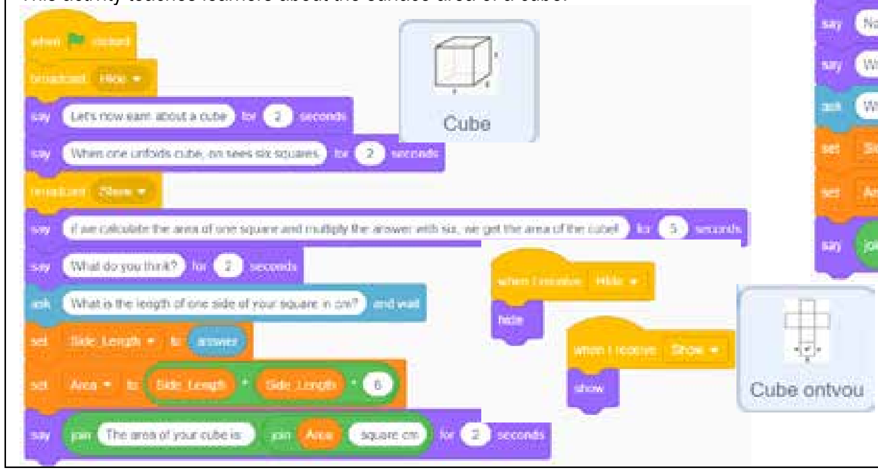
- Ciphers (pen-and-paper) – decrypt an encrypted message
- Microcontroller with microcontroller app, e.g., road signs, morse code, smileys, etc.
- Block-based coding app, e.g., an interactive story

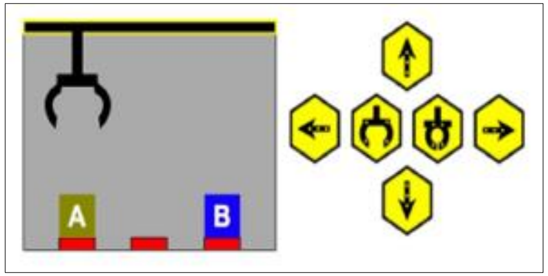
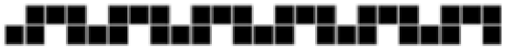
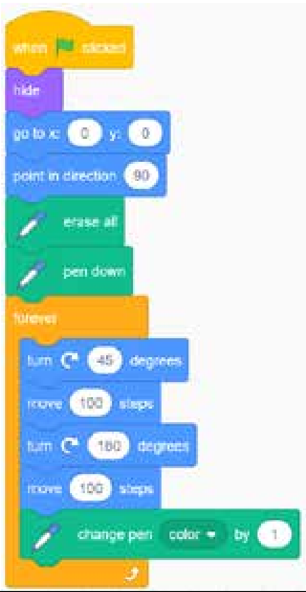
</td></tr>
</table>

Content (Grade 6 / Term 3)	Notes/Examples
D.9 Create a pattern to represent or communicate a message or image	**Link to C.2 – C.5, and R.5 – R.7**
Example activity: Simulate/display a simple message/ game (e.g., scrolling 'billboard message' or rock, paper, scissors game) on a microcontroller (LEDs on grid).	Learners need to represent or communicate a message or an image using • Ciphers (pen-and-paper) • Microcontroller with microcontroller app, e.g., road signs, morse code, smileys, etc • Block-based coding app, e.g., an interactive story
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.2 – C.5 and R.5 – R.7**
Reinforce and extend from the previous grades and terms using different examples and activities. Basic folder and file management and naming conventions	Integrate with Coding and Robotics activities done on the computer.

3.3.4 Term 4

Content (Grade 6 / Term 4)	Notes/Examples
Coding	
C.1 Apply computational thinking (CT) skills to develop a set of logical instructions to solve a problem.	**Link to C.1 – C.7 and R.5 – R.7**
Do with C.1 – C.7	**Broadcasting** is a way to send messages between sprites1 A broadcast is a message that is sent through the program, activating scripts with the matching hat blocks. With broadcast, users can broadcast some messages across the all the code (program). This illustrates the concept of parallelism in coding – when more than one action occurs at the same time/events that happen at the same time. This is an interesting feature that makes projects more interactive and creative.
C.2 Present a simple coding solution using symbolic or written statements representing sequences of commands, single repetition, and conditional constructs.	
Example activity 1 Broadcast and When I receive Provide learners with the program on the right. Let them run the program and watch what happens. Let them study the code and explain wat it does, Which sprite is broadcasting the message? Which sprite is receiving the message? **Example activity 2** Provide learners with a task/problem in where they need to use the Broadcast and When I receive. **Example activity 3 Open-ended** Leaners use their knowledge, skills and experience to write a program of their choice that uses broadcast and when I receive. Encourage them to also include a few other concepts learned. 	
C.3 Interpret and execute a given symbolic or written set of commands	**Link to C.1, C.2 and C.4, C.5**
Example activity 1 – Wait until Provide learners with the code on the right. Let learners run the code and describe what it does. Now, let learners explain what the *wait until* command does  **Example activity 2– Video motion** Execute the following code and watch what happens. (Move your hands in front of the PC camera) 	The wait until uses a condition (like an IF…THEN). **Note**; It is also advisable that learners create IPO tables as part of the planning to code a solution to a problem (also in coding C1,.1, C.2) e.g. 

Content (Grade 6 / Term 4)	Notes/Examples
Example activity 3 Study the code on the right and explain what it does. Change the code so that the sprite must only draw when it touches the mouse pointer  	
C.4 Debug a given symbolic or written set of instructions	**Link to C.1 – C.3**
Example activity Provide learners with the code on the right. Let them study the code and explain what the program does. Now, let them run the code and provide input as requested. The program does not provide the correct output. Learners need to find the bug and correct it.  **Extension activity** The following activity can be used as an extension to the activity on the left (Two sprites are used – Cube and Cube unfolded with broadcast) This activity teaches learners about the surface area of a cube.  	**Note:** Encourage learners to create a block-based program like the extension activity to teach younger learners something. The extension activity also provides an opportunity for learners to exhibit their skills to create sprites using a program such as Paint to import The extension activity is used to teach learners about the surface area of a cube (Learners can write Scratch code to 'teach' concepts to tother learners)

Content (Grade 6 / Term 4)	Notes/Examples
C.5 Evaluate a given solution towards potential improvement	**Link to C.1 – C.4**
Example activity 1 Provide learners with two programs that achieve the same goal, but in different ways (using different code) Let them study the code of both programs and explain the difference in code. Ask them if they think the one is 'better' than the other and why?	Learners need to evaluate code to determine which is the better code/more efficient code
C.6 Recognise and interpret patterns in symbolic sets of data or visualisations.	**Link to D.6 and D.7 and C.1**
Example activity 1 The crane in the port of Durban responds to six different input commands: 1. Left 2. Right 3. Up 4. Down 5. Grab 6. Release Crate A is in the left position, crate B is in the position on the right Which is the correct set of instructions to swap the position of the two crates? Write down the letter of the correct answer.  *https://olympiad.org.za/talent-search/past-papers/pen-and-paper/* A (Down, Grab, Up, Right, Down, Release, Up) B (Down, Grab, Up, Right, Down, Release, Up) (Right, Down, Grab, Up, Left, Left. Down, Release, Up) (Right, Down, Grab, Up, Right, Down, Release) C (Right, Right, Down, Grab, Up) (Left, Left, Down, Release, Up) D (Down, Grab, Up, Right, Right, Down, Release, Up) (Down, Grab, Left, Down, Release, Up) (Down, Grab, Up, Right, Down, Release, Up **Example activity 2** Learners write code to implement the algorithm of Term 3 C;6 Activity 1 (write code to draw the pattern)  **Example activity 3 Forever circle pattern** Provide learners with the code on the right and let them run the code. Now learners explain what the code does and discuss the pattern  	**Note**: Learners need to understand that when swopping two items, one needs an extra 'place' to keep one item temporarily. This concept is later used in programming when one wants to swop two variables.

Content (Grade 6 / Term 4)	Notes/Examples
C.7 Create or complete a pattern to represent a data set	**Link to D.6 and D.7 and C.1**

Example activity 1

Three spotlights are used to light the theatre stage, a red one, a green one and a blue one. The colour of the stage depends on which of the three spotlights are turned on.

The table below shows the possible combinations of colours.

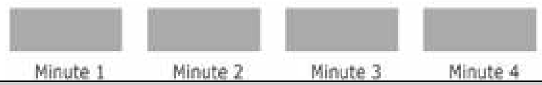
Red light	Green light	Blue light	Stage colour
off	off	off	Black
off	off	on	Blue
off	on	off	Green
off	on	on	Cyan
on	off	off	Red
on	off	on	Magenta
on	on	off	Yellow
on	on	on	White

https://olympiad.org.za/talent-search/past-papers/pen-and-paper/

From the beginning of the show, the lights will be switched on and off in the following pattern:

- The red light repeats the sequence: two minutes off, two minutes on.
- The green light repeats the sequence: one minute off, one minute on.
- The blue light repeats the sequence: four minutes on, four minutes off.

What will the colour of the stage be in the first 4 minutes of the show



Robotics	
R.1 Explain what a robot is in simple terms.	**R.1 – R.4 can be done together**
R.2 Identify different types of robots.	Revise and extend from previous grades and terms and extend to application of artificial intelligence (AI) in robotics
R.3 Outline the different components of a robot	
R.4 Present an understanding of how robots affect the world	

Example activity – How AI is applied in robotics

AI is applied in robotics to make robots smarter and more capable. Here's how AI is used in robotics:

Perception: AI helps robots understand and perceive the world around them. Sensors, cameras, and other devices provide information to the robot, and AI algorithms analyse and interpret that data. This allows robots to recognize objects, understand speech, detect obstacles, and navigate their surroundings.

Learning and Adaptation: AI enables robots to learn from experience and improve their performance over time. Machine learning algorithms, a type of AI, allow robots to gather data, identify patterns, and make predictions. By learning from their interactions, robots can adapt their behaviour, refine their skills, and become more efficient in completing tasks.

Decision-Making: With AI, robots can make decisions based on the information they gather and analyse. They can evaluate different options, consider factors such as safety and efficiency, and choose the best course of action. AI algorithms help robots reason, plan, and make intelligent choices, allowing them to perform complex tasks autonomously.

Human-Robot Interaction: AI enables robots to interact with humans in more natural and intuitive ways. Natural language processing allows robots to understand and respond to voice commands, while computer vision enables them to recognize facial expressions and gestures. This makes it easier for people to communicate and collaborate with robots, opening possibilities for assistance, companionship, and teamwork.

Notes/Examples:

The input to the robot will be via sensors and transducers. The various sensors used in robotics include:

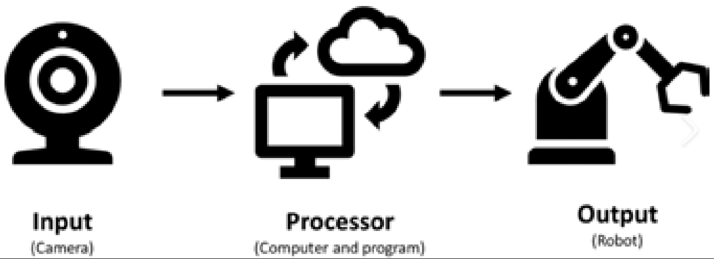
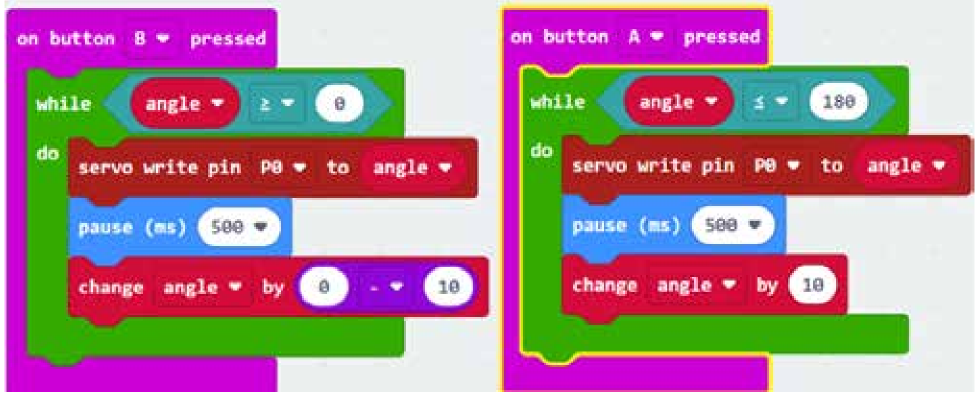
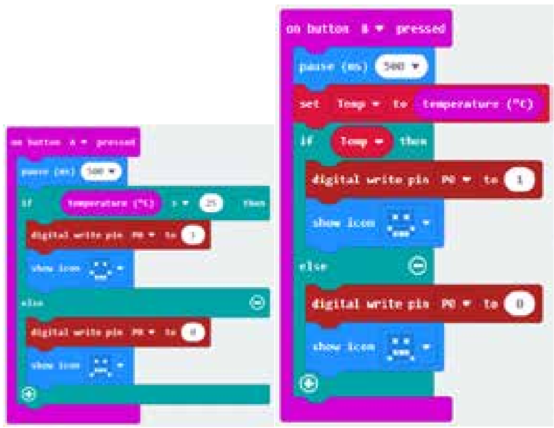
- Contact/touch sensors
- Temperature sensors
- Light sensors
- Sound sensor
- Proximity sensor
- Distance sensor
- Pressure sensor, etc.

Input

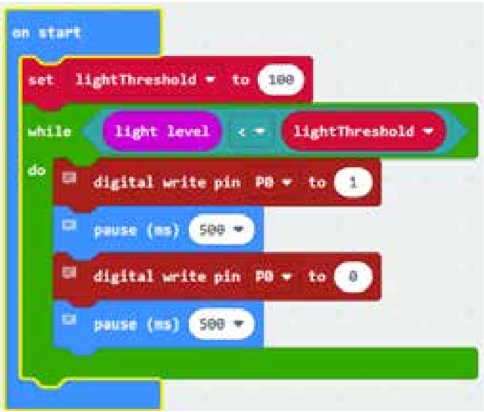
Based on the type of the sensor output, digital or analogue, the further circuitry is decided.

Output

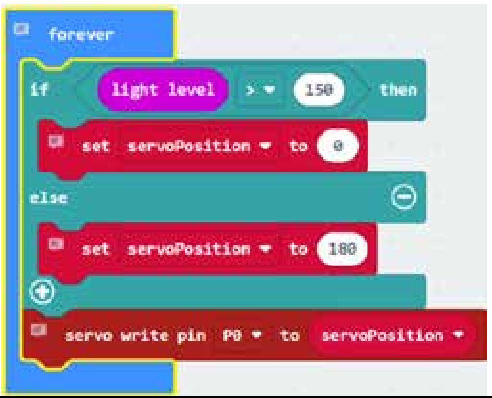
The output of the robot will vary according to its driving load. The most common output units are:

Content (Grade 6 / Term 4)	Notes/Examples
Autonomous Operation: AI empowers robots to operate autonomously, meaning they can perform tasks without constant human supervision. Robots equipped with AI can analyse their environment, make decisions, and execute actions independently. This autonomy allows robots to be more efficient, work in challenging environments, and carry out tasks that would otherwise be difficult or dangerous for humans. By combining AI with robotics, we can create intelligent machines that can perceive, learn, reason, and interact with the world around them. This integration enables robots to perform a wide range of tasks, from manufacturing and healthcare to exploration and assistance, making our lives easier and advancing technology to new frontiers. **Example activity 2** Learners create a concept map of a robot that includes what it a robot is with the different components with their functions (See Annexure A) 	• Actuator • Relays • Speakers • CD screens, etc. based on the applications the carrier changes. For wired applications, cables and wires are used while for wireless robots RF, RFID, Wi-Fi, DTMF, etc., technologies are used. **Processing** As far as processing is concerned, a microcontroller or microprocessor can be used. This choice will depend on the driving load. A microcontroller is cheap and is easy to program than a microprocessor. However, a microcontroller has very low output power, and so cannot drive large loads. On the other hand, its PC counterpart, the microprocessor can drive large loads at its output. The complexity of the operations being performed by the robot will depend on its processing unit.
R.5 Design a simple artefact based on a set of design instructions	**Link to R.6 – R.7 and C.1 – C.7**
Example activity 1 – Servo incremental movement with loop Use while loops and variables to control a servo motor's incremental movement. Coding to make the servo motor move in increments of 10 degrees from 0 to 180 when button A is pressed and then back to 0 when button B is pressed  **Example activity 2 – Make an LED blink continuously until the room becomes sufficiently bright**	Introduction to variables should be done on a very gradual manner. In Grade 6 term 4, only the concept of a single variable relating to the (answer block in Scratch) should be introduced.  The learners should be introduced to variables where the variable has a similar name than that of the sensor value read. An analogy of the sensor value being the same as the answer block is then used. The answer is then assigned to a variable with an equivalent name. One variable only!

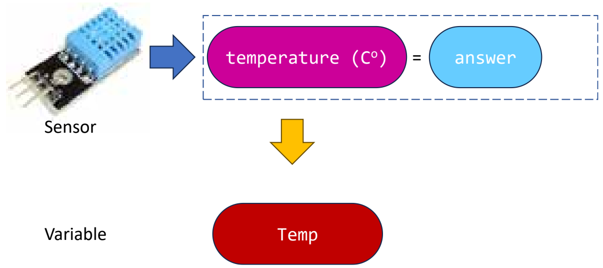
Content (Grade 6 / Term 4)



Example activity 3 – Basic light-controlled Servo
Move the servo to a position based on the light level, using a variable to store the servo's position.



Notes/Examples



Note: Sensor refers to the sensor on the Microcontroller.

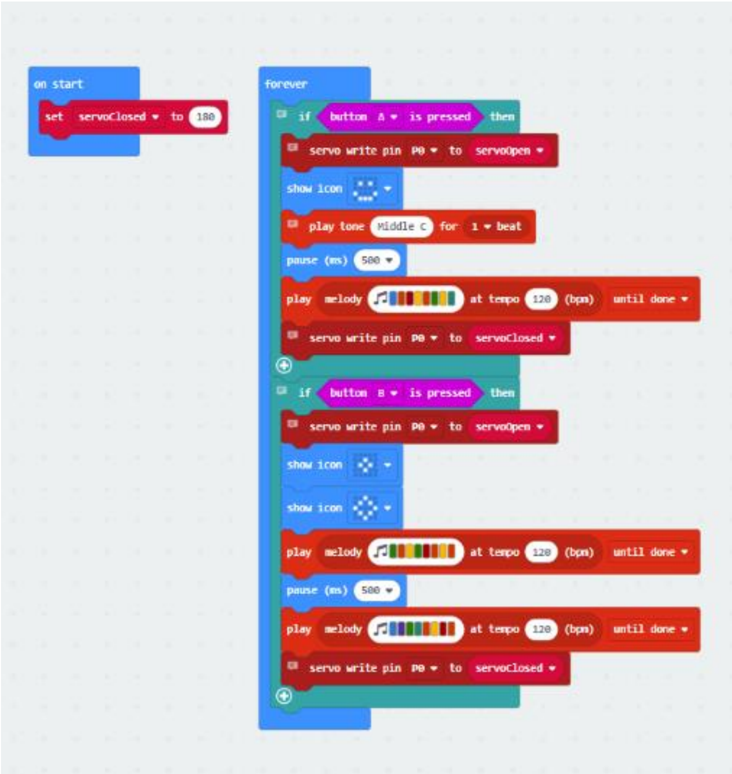
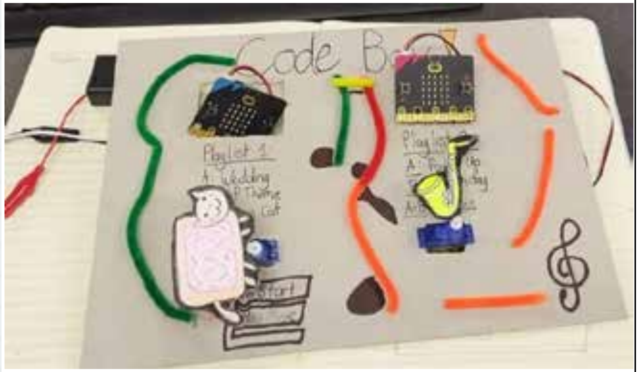
R.6 Mimic the operations of a robot

Example 1 Flip-a-coin simulator
The microcontroller must simulate flip-a-coin. Decide which picture you want to display for heads and for tails.
Possible solution: Choose a random number between 0 and 1. If 0, display "head" if 1, display "tail".)

Example 2 Pet mood detector
Shake the microcontroller to generate a random number that will indicate the mood or state of your pet. Display the pet's mood or state.
Number 0: Pet is hungry
Number 1: Pet is happy.
Number 2: Pet is tired.
Number 3: Pet wants to play.
Number 4: Pet is scared.
Number 5: Pet loves you.

Link to R.5 and C.1 – C.7

Learners can create various programs in a block-based programming environment

Content (Grade 6 / Term 4)	Notes/Examples
R.7 Create, test and execute a set of robotic instructions	**R. 6 and R.7 are done together with R.5 (once enabling activities in R.5 are completed) to complete the project**
Project – Automated System: Sound-Controlled Friendly Monster Demonstrate how servos, sensors, and microcontrollers can interact to create automated systems. **Project - Musical Memory Box** Design and assemble a musical memory box using a Microcontroller and a servo motor. Each compartment of the box should play a distinct tune when revealed.  	
Digital Concepts	
D.2 Recognise that he or she is living as a citizen in a digital world.	**Link to D.6**
Example activity: "Digital Devices Dilemma" Case Study: ***Introduction***: In this case study, we will explore the story of three friends, Alex, Mia, and Jake, who are in the sixth grade. They each have different approaches to using digital devices, and their experiences will help us understand the importance of responsible and balanced use of technology. ***Characters***:	**Aspects addressed** • Responsible use of digital devices involves balancing screen time with other activities, such as schoolwork, physical activities, and spending time with family and friends. • Overusing digital devices, especially late at night, can negatively affect sleep patterns and overall well-being.

Content (Grade 6 / Term 4)	Notes/Examples
• Alex: Enjoys spending a lot of time on digital devices, especially playing video games and watching online videos. Alex often stays up late at night using a smartphone or tablet. • Mia: Uses digital devices for schoolwork and educational purposes, such as research and online learning. She also likes to connect with friends and family through social media. • Jake: Tries to limit his screen time and uses digital devices mainly for schoolwork or when necessary. He enjoys spending time outdoors and engaging in sports and hobbies. ***Scenario***: One weekend, Alex, Mia, and Jake all have a project to complete for their science class. They need to research different animal species and create a presentation about their findings. They also want to use their devices to have some fun during their free time. Actions and Consequences: ***Alex's Approach***: Alex spends most of the weekend playing video games and watching YouTube videos. As a result, Alex does not have enough time to work on the science project until the last minute. On Monday morning, Alex is stressed and unprepared for the science class presentation, which affects the overall quality of the project. ***Mia's Approach:*** Mia uses digital devices responsibly and allocates specific time for research, completing her project, and connecting with friends and family. She finds reliable online sources for her research, which adds depth to her science project and impresses her teacher with well-structured content. ***Jake's Approach:*** Jake balances his time between outdoor activities, schoolwork, and digital device use. He spends some time researching for the science project, but he also makes sure to get enough rest and engage in physical activities during the weekend. Jake delivers a well-prepared presentation that showcases his knowledge and creativity. **Example Discussion Questions:** • How did Alex's excessive use of digital devices impact his performance on the science project? • How did Mia's balanced approach benefit her in completing the project effectively? • What positive habits did Jake demonstrate in his use of digital devices, and how did it contribute to his overall well-being? **Example activity 2:** Understanding living as citizens in a digital world (Dangers, screen time, etc.) Learners watch the following video: Film Mesir "L'altra par" #2 - YouTube After watching, learners write down one problem identified in this video and how to overcome this problem. Have a class discussion about what is identified in the video, how society can overcome these problems, which of these problems learners are experiencing, etc.	• Using digital devices for educational purposes can be beneficial, but it's essential to set time limits and stay focused on the task at hand. • A balanced approach to using digital devices can lead to better time management and improved performance in school and other activities. • Screentime
D.3 Demonstrate an understanding of the concept of a computing device.	**Link to D.1, D.4 and D.5**
Example activity 1 Learners complete a concept map of a computing device: What it is, components and functions, etc. (See Annexure A)	Done with D.1
D.4 Identify the common uses of ICT in the real world	**Link to D.1 and D.5**
Example activity 1: Embracing the Digital Revolution: The Impact of ICT on Work and Life Use a case study to revise. Introduction: Information and Communication Technology (ICT) has revolutionised the world we live and work in, transforming various industries and daily activities. This case study explores the common uses of ICT in the real world and delves into the profound impact it has had on society, both professionally and personally. Common Uses of ICT in the Real World: ICT has found its application across multiple sectors, enhancing productivity, communication, and access to information. The case study examines how ICT is utilised in various areas, such as:	Reinforce and extend from the previous grades and terms using different examples and activities.

Content (Grade 6 / Term 4)	Notes/Examples
a. Communication: The widespread use of smartphones, email, social media platforms, and video conferencing has revolutionized the way people communicate globally. ` b. Business and Industry: ICT has streamlined business processes by allowing communication, sharing and collaboration. c. Education: E-learning platforms, digital libraries, and interactive educational tools have transformed the way learners access and engage with knowledge. d. Healthcare: ICT has enabled telemedicine, remote patient monitoring, electronic health records, and medical research advancements. e. Entertainment: Online streaming services, gaming platforms, and virtual reality experiences have reshaped the entertainment industry. Questions: a. How has ICT enhanced communication in personally and in business? b. In what ways has the integration of ICT improved productivity in businesses?	
D.5 Differentiate between the components of an ICT system	**Link to D.4**
Example activity Provide learners with a picture diagram indicating of a common ICT system (e.g. basic point-of-sales) Let them label the components and briefly describe the role/function of each within the system Discuss the picture diagram with the class and let learners correct what they might have misinterpreted or got wrong.	
D.6 Explain how the adaptation of technology impacted the world we work and live in	**Link to D.2**
Combine with case study D.1.	Done with D.4
D.7 Present a basic understanding of the concept of input processing and output.	**Link to D.3, C.1-C.5 and R.5-R.7**
Link to concept map activity in D.3	Done with D.3
D.10 Demonstrate a basic proficiency in the application of digital skills.	**Link to C.2 – C.5 and R.5 – R.7**
Reinforce and extend from the previous grades and terms using different examples and activities. Basic folder and file management and naming conventions. Use Paint to create sprites and backgrounds to import to block-based coding app.	Do with Coding and Robotics activities done on the computer.

4 SECTION 4 ASSESSMENT

4.1 ASSESSMENT

Assessment is a continuous planned process of identifying, gathering and interpreting information about the performance of learners, using various forms of assessment. It involves four steps: generating and collecting evidence of achievement, evaluating this evidence, recording the findings, and using this information to understand and thereby assist the learner’s development to improve the process of learning and teaching.

Assessment involves activities that are undertaken throughout the year. Assessment comprises two different but related activities: informal daily assessment (assessment for learning) and formal assessment (assessment of learning).

Assessment in Coding and Robotics should encourage computational thinking practices, i.e. integrating the power of human thinking with the capabilities of ICTs and computer programming.

However, cognisance should also be taken of what is being assessed. Certain competencies are best assessed with particular forms of assessment. Different kinds of assessments are appropriate to the competencies necessary for different topics at different age groups. It is useful to use an observation checklist to assess learners measuring in the early grades. Rubrics, for example, can be used to evaluate learner’s Coding and Robotics as well as problem solving skills.

Assessment involves activities that are undertaken throughout the year. In grades 4 – 6 assessment comprises two different but related activities: informal daily assessment (assessment for learning) and formal assessment (assessment of learning).

Assessment in Coding and Robotics should encourage computational thinking practices, i.e. integrating the power of human thinking with the capabilities of ICTs and computer programming.

4.1.1 Informal or daily assessment

Assessment for learning has the purpose of continuously collecting information on a learner’s achievement that can be used to improve their learning. Informal assessment is the daily monitoring of learners’ daily progression and should also focus on how learners learn and retain new information. It should therefore include retrieval practice (as described by the science of learning –section 2.7.5) as well as deliberate practise (See Section 2.7.4).

Trying to remember something enhances memory, and teachers can use quizzes or self-tests for this purpose. As learners learn and retain new information by focusing on the meaning of the content, teachers can assign tasks that require learners to explain or organise the material (e.g. concept maps), which helps them think about the meaning of content.

In learning Coding and Robotics, practise is also essential, and teachers can focus on regular practise and retrieval as well as spaced practise and retrieval over time to aid long-term retention. Teachers can also interleave different types of practice and use multiple modalities to enhance learning

Informal assessment and retrieval practise may be as simple as stopping during the lesson to ask questions or have learners writing down what they can remember about what was learned in a previous lesson and provide feedback to the learners. Informal assessment does not need be recorded. It’s part of all learning activities taking place in the classroom. Learners or teachers can mark these tasks.

Self-assessment and peer assessment actively involves learners in assessment. This is important as it allows learners to learn from and reflect on their own performance. The results of the informal daily assessment tasks are not formally recorded unless the teacher wishes to do so. The results of daily assessment tasks are not used for promotion and certification purposes.

4.1.2 Formal assessment

All assessment tasks that make up a formal programme of assessment for the year are regarded as formal assessment. Formal assessment tasks are marked and formally recorded by the teacher for progression and certification purposes. All formal assessment tasks are subject to moderation for the purpose of quality assurance, and to ensure that appropriate standards are maintained.

Formal assessment provides teachers with a systematic way of evaluating how well learners are progressing in a grade and in a particular subject. Examples of formal assessments include tests, examinations, practical tasks, projects, etc. Formal assessment tasks form part of a year-long formal programme of assessment in each grade and subject.

The following tables provide the formal assessment requirements for Coding and Robotics:

Table 4-9 Minimum formal assessment requirements for Coding and Robotics

	Forms of assessment	Minimum requirements per term				No of tasks per year	Weighting
		Term 1	Term 2	Term 3	Term 4		
SBA	Tests	1		1			**75%**
	Examination		1		1		
	Practical Tasks	1	1	1			
	Project				1		
	Total	**2**	**2**	**2**	**2**	**8**	
End-of-year examination							**25%**

The forms of assessment used should be age and developmental level appropriate. The design of these tasks should cover the content of the subject and include a variety of tasks designed to achieve the objectives of the subject.

4.2 PROBLEM-BASED LEARNING

Assessment in Coding and Robotics can be done assessing the learner in action, for example, watching the learner solving the problem without stopping the moment. This can be done using the following strategies:

4.2.1 Individual Problem-based Learning (coding)

Problem solving is the process of designing, evaluating, and implementing a strategy to answer question, complete a task or achieve a desired goal.

4.2.1.1 Types of problems

In terms of coding, typically, problems could require learners to:

- provide missing code instructions (code instructions are provided with some instructions or code elements missing / to be completed or
- choose the correct solution from 2-3 options or
- work through (trace) / act out code to determine if it is correct and correct if required or
- rewrite a set of coding instructions to be more efficient or
- compare different solutions to evaluate efficiency or
- translate verbal/written instructions to code (e.g. packing arrows)
- develop the solution/algorithm (code instructions) themselves using computational thinking and following problem-solving process.

The above will depend on the competency the learner needs to demonstrate. Coding problems need to gradually increase in terms of complexity.

4.2.1.2 Assessing problem-based learning (coding)

The learner is assigned a problem he/she must solve and in doing so:

- needs to understand the problem.
- analyses the problem (what is given and what is needed / what is important and what can be ignored - abstraction).
- identifies the main steps (abstraction / high level solution).
- identifies the detailed steps (decomposition / breaking down the main steps).

- Identifies patterns to determine the need for using coding structures such as repetition.
- implements and tests the solution (algorithm).
- debugs the solution if required.

Refer to Annexure B for rubric example to assess problem solving.

4.2.2 Cooperative Learning

Instead of encouraging learners to compete for grades or achievement, cooperative learning asks them to work together and participate in group learning activities (small groups, e.g. 4 learners), under the guidance of a teacher.

Assessing cooperative learning in Intermediate Phase Coding and Robotics

Example rubric to assess cooperative learning activity: *Defining a robot and its different parts*.

Refer to Section 2.7.2 for example cooperative learning activity.

Refer to Annexure B for rubric example to assess cooperative learning.

4.2.3 Pair Programming

Assessing pair programming in Intermediate Phase Coding and Robotics

Example rubric to assess cooperative learning activity:

Identifying, completing and creating patterns.

Refer to Section 2.7.3 for example pair programming learning activity.

Refer to Annexure B for rubric example to assess pair programming.

4.3 Recording and Reporting

Recording is a process in which the teacher documents the level of a learner's performance in a specific assessment task. It indicates learner progress towards the achievement of the knowledge as prescribed in the Curriculum and Assessment Policy Statements. Records of learner performance should provide evidence of the learner's conceptual progression within a grade and her / his readiness to progress or being promoted to the next grade. Records of learner performance should also be used to verify the progress made by teachers and learners in the teaching and learning process.

Reporting is a process of communicating learner performance to learners, parents, schools, and other stakeholders. Learner performance can be reported in several ways. These include report cards, parents' meetings, school visitation days, parent-teacher conferences, phone calls, letters, class or school newsletters, etc. Teachers in all grades report in percentages against the subject. The various achievement levels and their corresponding percentage bands are as shown in the Table below.

RATING CODE	DESCRIPTION OF COMPETENCE	PERCENTAGE
7	Outstanding achievement	80 – 100
6	Meritorious achievement	70 – 79
5	Substantial achievement	60 – 69
4	Adequate achievement	50 – 59
3	Moderate achievement	40 – 49
2	Elementary achievement	30 – 39
1	Not achieved	0 - 29

4.4 General

This document should be read in conjunction with:

- National policy pertaining to the programme and promotion requirements of national Curriculum statement Grades R-12; and
- The policy document, National Protocol for Assessment Grades R-12

ANNEXURE A: TERMINOLOGY

The following tables provide clarity on some terminology used in the CAPS

A.1 CODING

Table A-10 Coding - Clarification of concepts and terms

Term/Concept	Explanation
Algorithm	An algorithm is a set of logical instructions/commands that a human or computer can execute to solve a specific problem or accomplish a particular task. It is a computational process that uses a finite number of steps (logical instructions or commands), carried out in a specific sequence to solve a problem.
Coding	**Coding** is the process of writing instructions that a computer can understand and execute These instructions are written in a programming language, which is a set of rules that define how the instructions should be written. The purpose of coding is to create software programs that can perform specific tasks, such as running a website, playing a video game, or analysing data.
Computation	In computing, **computation** refers to any type of arithmetic or non-arithmetic calculation that is well-defined. It can involve mathematical equations, computer algorithms, and other types of calculations.
Computational Thinking	It refers to a problem-solving approach that involves breaking down complex problems into smaller, more manageable parts and using algorithms and logical reasoning to solve them. It involves skills such as abstraction, decomposition, pattern recognition, and algorithmic thinking. It is a way of thinking that is used in computer science, but it can also be applied to other fields. In education, computational thinking is used to teach learners how to think logically and solve problems systematically.
Conditional (choice/ decision) statement	A control structure that selects one alternative from two or more possible execution sequences to be executed
Control statement	A control structure that is used to modify the order in which instructions are executed such as a loop or conditional statement
Event	A signal or notification that something has happened.
Expression	Refers to a combination of one or more values, operators that can be evaluated to produce a result.
Input	In computing, input refers to the data that is entered into a computer system, such as text, images, or sound,
IPO table	Input-Processing-Output table describes the inputs processing and outputs of program.
Loop statement	A control structure that allows a sequence of instructions to be continually repeated until a certain condition is reached
Operator	**Operators** are symbols or keywords that represent computations or actions performed on operands. Operators include: Arithmetic operators (+, -, x, /, modulo), comparison operators (=, >, <, ≤, ≥, ≠), Boolean operators OR, AND, NOT, string operators for manipulating strings/text (length, concatenate, indexing) Operators provide the building blocks for creating expressions and performing operations
Output	In computing, output refers to the result of the processed data that is presented to the user in a usable format. This can be in the form of text, sound, image, or video.
Processing	In computing, processing refers to the operations performed by the computer to manipulate or analyse the input data.
Program	A **program** is a sequence of instructions that a computer can execute to perform a specific task.
Trace table	In programming, a trace table is a technique used to test an algorithm and predict step by step how the computer will run the algorithm. Statements are executed step by step, and the values of variables change as an assignment statement is executed. A trace table simulates the flow of execution by showing the values of variables at each step of the algorithm. Trace tables are typically used by novice programmers to understand how an algorithm works and to identify errors in the algorithm 2
Variable	In programming, a variable is a named storage location that holds a value or data. Variables are essential for storing and manipulating data in computer programs. The values in variables can change during the execution of a program.

A.2 Robotics

Table A-11 Robotics - Clarification of concepts and terms

Term/Concept	Explanation
Actuator	Refers to a device that converts energy into physical motion, such as rotation or translation. Actuators are often called the muscles of robots, as they enable robots to perform various tasks and interact with the environment
Controller	Refers to a device that commands, directs, and regulates the behaviour of a robotic system. It takes input signals from the robot's sensors, processes them based on programmed instructions, and then sends output signals to the robot's actuators to perform the desired actions.
Microcontroller	Refer to a type of small computer that can control the functions and behaviour of a robotic system. It generally consists of a processor, memory, input/output ports and other peripherals that can be programmed to perform specific tasks. It can receive data from sensors, process it according to the programmed instructions and send commands to actuators.
Robot	A **robot** is a machine that can perform a series of actions automatically, either by being programmed by a computer or by being guided by an external control device.
Sensor	Refers to a device that can measure or detect some physical property of the environment or the robot itself and convert it into an electrical signal. Examples include light sensor, touch sensor, sound sensor, etc.

A.3 Digital Concepts

Table A-12 Digital Concepts - Clarification of concepts and terms

Term/Concept	Explanation
Cipher	A cipher, also known as an encryption algorithm, is a set of well-defined rules used to transform information into a scrambled form, called ciphertext. It is used to encrypt messages so that they can only be read by someone who knows how to decrypt them.
Computing device	A general-purpose machine that can execute instructions for any data processing purpose. A computing device can receive input, do something with the input and provide a result or output.
Data	Raw, unprocessed facts and figures.
Decode	Reconstructing the original **(encoded)** information. It involves taking an encoded representation and converting it back into its original form
Decrypt	The reverse process of encryption, taking ciphertext and using the appropriate key to convert it back into its original, readable plaintext form.
Digital Citizen	A person who uses the Internet and other digital technology to communicate with other and engage in society.
Digital Citizenship	The ability to participate in online society. It includes concepts like respecting others' privacy, avoiding cyberbullying, netiquette, digital health and welfare, ability to assess the credibility and reliability of online information, intellectual property, impact and responsibility of online actions and deeds.
Digital Footprint	The trail of traceable digital activities, actions, contributions, and communications one leaves behind when using the Internet or digital devices.
Encode	Converting information into a specific format (transforming data or messages into another format)
Encrypt	The process of transforming readable data (plaintext) into an unreadable, scrambled form (ciphertext) using a cryptographic algorithm (cipher) and a secret key.
Hardware	The physical building blocks of a computing device or the tangible parts you can see and touch. It includes: • **Central Processing Unit** (CPU): the component responsible for executing instructions. • **Random Access Memory** (RAM): Component for temporary storage of programs and data the computing device is currently working with. • **Storage devices**: E.g. hard drives, solid-state drives (SSDs), for permanent data storage. • **Input devices** such as keyboard, mouse, screen, microphone mouse, used to interact with the computer. • **Output devices** such as screen, speakers, printer, etc., used to display and output information.
Information	Data that has been processed and organised to convey meaning.
Information and Communications Technology (ICT)	ICT is the use of computing and telecommunication technologies, systems, and tools to facilitate the way information is created, collected, processed, transmitted, accessed and stored

Information Technology (IT)	IT refers to the use of computer systems to manage, process, protect, and exchange data and information.
Input	In computing, input refers to the data that is entered into a computer system, such as text, images, or sound.
Output	In computing, output refers to the result of the processed data that is presented to the user in a usable format. This can be in the form of text, sound, image, or video.
Personal information	In computing, **personal information or personal data** is any information or data that can identify a person, from one's name and address to one's device identifier and account number.
Processing	In computing, processing refers to the operations performed by the computer to manipulate or analyse the input data. This includes executing software applications, performing calculations, sorting and filtering data, and running programs.
Software	The intangible programs and applications (instructions) that give life to the physical components. Examples include: • **Operating System** (OS) that manages the hardware resources and provides a platform for running other programs. (e.g., Windows, macOS, Linux) • **Application software**: Specific programs designed for performing tasks like word processing, image editing, games, etc. • **Programming languages** used to create new software by writing instructions the computer can understand.
Technology	Encompasses any tool, technique, or process used to solve problems and manipulate our environment. Technology is designed with a purpose of solving problems that meet human needs and wants. It refers to tools, machines, or devices that make our lives easier or better.

Annexure B: Assessment Examples

B.1 Cooperative Learning

Example rubric to assess cooperative learning activity: *Defining a robot and its different parts*.

	Learner name	#Definition of robot	#Flashcards utilised well.	#Drawing illustrates robot	*Learner fulfilled role well
1.					
2.					
3.					
4.					

#Replace with suitable criteria depending on the task/problem
*Will remain the same irrespective of task/problem

Note:
Although all learners in the group get the same mark for the first three criteria, each learner gets an individual mark for the "Learner fulfilled role well" – this is based on how well each learner contributed based on their set role.

The teacher can give mark these while learners are completing the activity and hence it should not require much extra time.

Each of the aspects listed in the table above, could be assessed using the following example:

Aspect assessed	Beginning (1)	Developing (2)	Accomplished (3)	Exemplary (4)
				
Definition of concept, e.g. robot	Key information is missing (e.g. no parts included) and the definition is unclear and difficult to follow	Some key information is included, and the definition is generally clear and easy to follow but may be incomplete or somewhat disorganised.	Most of the key information is included (e.g. most of the parts) and it is mostly well-organised and easy to follow	The learner demonstrates full understanding in that the definition is well-organised, complete, and easy to follow.
Flashcard utilised well	Flashcards are not used effectively	Some attempt is made to use the flashcard to explain the concept, but it lacks detail and key information	Flashcards are used appropriately to explain the concept and includes most of the key information	Flashcards used effectively/innovatively to support a complete explanation of the concept and all key information
Drawing illustrates concept, e.g. robot	Drawing attempts to convey the concept, but the drawing is incomplete and/or difficult to interpret	Drawing includes some relevant details that may not all be accurate and conveys the concept but lack detail	Drawing includes most of the relevant and accurate details that appropriately convey the concept	drawing includes rich, and accurate details that effectively convey the concept.
Learner fulfilled role well	Learner does not understand his/her role and makes no contribution or unrelated contributions	Shares ideas or tries to fulfil her/his role, but does not work with group and most of the contributions are unrelated	Tries to understand his/her role and mostly makes relevant contributions. Can work on her/his part and take part in the group	Generates ideas and builds upon other's ideas to develop a larger plan. Works independently to do his/her part and is invested in the other group members (e.g. helps when needed, cares about the group product)

B.2 Pair Programming /Completing a Task in Pairs

Example rubric to assess pair programming activity: *Identifying, completing and creating patterns*.

	Learner name	#Concept1	#Concept2	#Concept2	*Learner fulfilled role well
1.					
2.					

#Replace with suitable criteria depending on the task/problem
*Will remain the same irrespective of task/problem

Note:

Although both learners get the same mark for the first three criteria, each learner gets an individual mark for the "Learner fulfilled role well" – this is based on how well each learner contributed based on their set role.

The teacher can give most of these marks while learners are completing the activity and hence it should not require much extra time.

Each of the aspects listed in the table above, must be assessed using a rubric:

B.3 Design Thinking

A process that emphasizes creativity, experimentation, and iteration to arrive at the best solution that meets user needs.

Competencies	Beginning (1)	Developing (2)	Accomplished (3)	Exemplary (4)
Inspiration: Learner applies creative thinking to create a product or complete a task	• Demonstrates limited creative thinking and understanding of the problem or task	• Applies creative thinking to understand the problem or task and identifies some opportunities for innovation	• Applies creative thinking effectively to gain a deeper understanding of the problem or task and identifies significant opportunities for innovation.	• Demonstrates exceptional creative thinking and in-depth understanding of the problem or task, uncovering unique insights and opportunities for innovation
Ideation: Learner can create own ideas to create a product or completing a task.	• Unsure about what is expected so any idea is scattered or unfocused and ideas do not clearly connect to the problem or task.	• Generally, mimics ideas from others (rather than creating new ideas) that are related to the problem or task.	• Creates new ideas that include enough detail and that are directly related to the problem or task.	• Creates many clear ideas by considering lots of possibilities that focuses on key information and fully addresses the problem or task
Implementation: Learner can use best ideas to create a product or complete a task.	• Creates a product or performance, but the product has limited functionality or detail and does not clearly address the problem, or the product is not useful.	• Creates a product or performance with some functionality that is somehow related to the challenge or problem.	• Uses ideas to create a product or performance with good functionality that is directly related to the problem or task.	• Creates clear ideas to create a product or performance with precision and full functionality and that fully addresses the problem or task.
Testing & Improving	• Provides minimal or no feedback and does not reflect on the quality to consider improvements or iterations	• Collects some feedback and reflects somewhat on the quality for considering minor improvements or iterations	• Collects thorough feedback, reflects accurately on the quality to inform improvements, and iterates on the solution	• Collects extensive feedback, conducts rigorous testing, and iterates on the design or solution based on feedback, leading to transformative improvements.

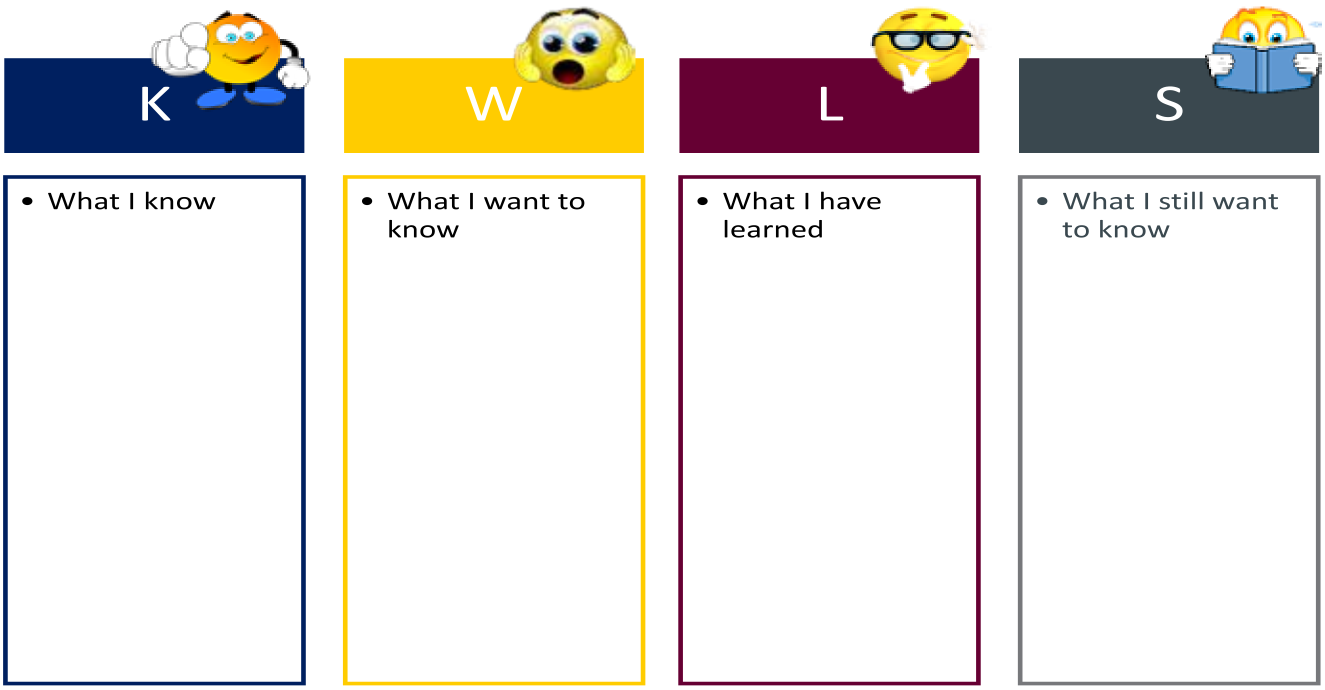
Note: All rubrics serve as examples only and may be adapted

Annexure C: Teaching Resources

C1 KWLS Chart

The KWLS chart is a learning strategy that helps learners engage with a topic in a structured and reflective manner. The chart helps learners organize their thoughts and track their progress as they explore a particular topic or concept.

The KWLS chart is a valuable tool for learners of all ages and levels of education. It promotes active engagement with the learning material, fosters critical thinking and inquiry, and supports metacognitive skills development. By using the KWLS chart, learners become more self-directed and aware of their learning process, leading to a more enriched and effective learning experience.



K - What I Know: In this section, learners write down what they already know about the topic. This step helps them activate their prior knowledge and make connections with the new information they are about to encounter. Identifying what they already know also helps learners build a foundation for further learning and enables them to understand how the new information fits into their existing knowledge framework.

W - What I Want to Know: In this part, learners jot down questions or areas of interest they have about the topic. These are the aspects they hope to learn more about or understand better as they engage with the subject matter. This step encourages curiosity and sets the stage for active exploration. By noting down their questions, learners become more focused and motivated to seek answers and engage with the learning materials more critically.

L - What I Have Learned: As learners progress through their learning journey, they record the new information, insights, and understanding gained about the topic. This section allows learners to summarise and consolidate their learning experiences. It reinforces the concepts they have grasped and helps them reflect on the new knowledge acquired. Reflecting on what has been learned enhances comprehension and retention of the material.

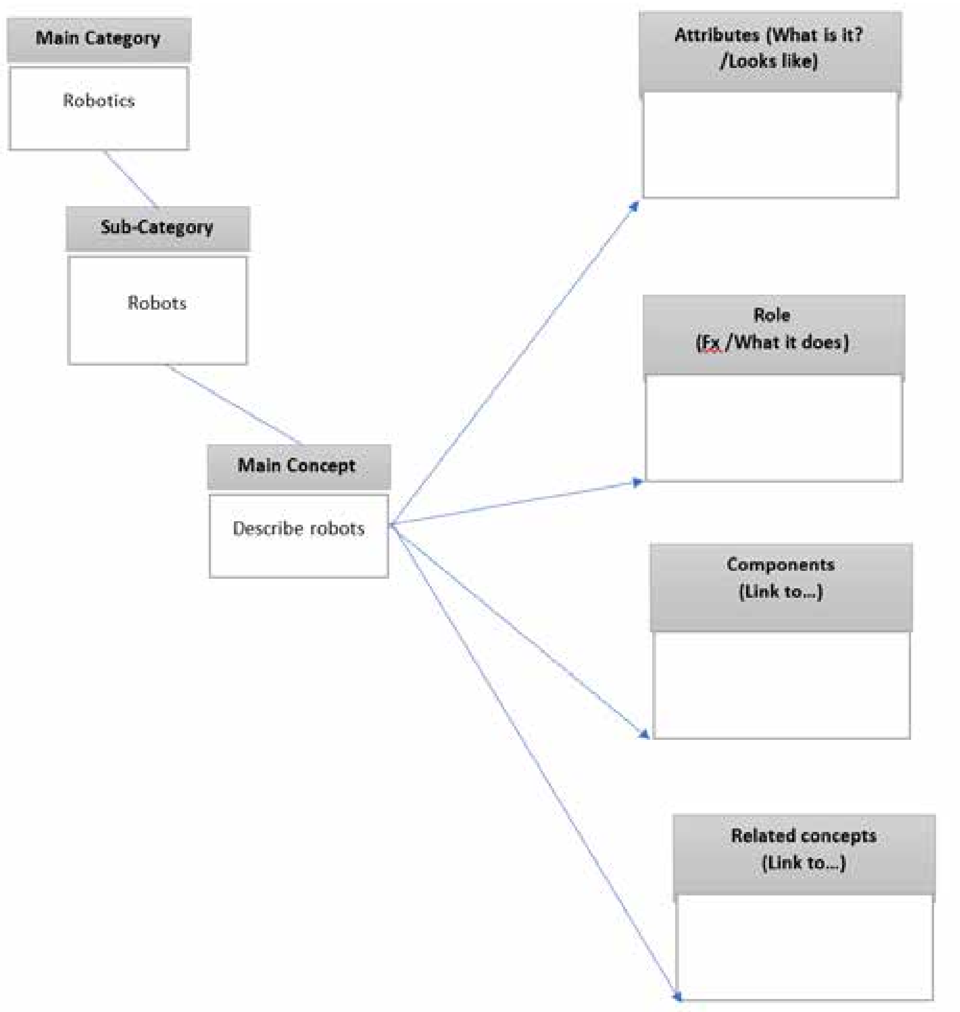
S - What I Still Want to Learn: In the last section, learners identify any remaining questions, uncertainties, or areas they would like to explore further. Even after learning a considerable amount about the topic, learners may realize that some aspects still require clarification or deeper investigation. This step encourages a growth mindset, as learners recognize that learning is an ongoing process, and there is always more to discover.

C2 CONCEPT MAPS

A concept map is a diagram that shows the relationships between different ideas. This helps you understand how they're connected. Every concept map — whether it's simple or complex — is made up of two key elements:

- **CONCEPTS:** These are typically represented by circles, ovals, or boxes and are called "nodes."
- **RELATIONSHIPS:** These are represented by arrows that connect the concepts, and the arrows often include a connecting word or verb (but they don't have to). These arrows are called "cross-links."

Example of a simple high-level concept map for understanding robots



Other resources to be considered:

https://www.teachwithict.com/
http://code-it.co.uk/gold/
https://sites.google.com/gshare.blackgold.ca/blackgoldmicrobit/microbit
https://www.instructables.com/
https://www.101computing.net/bbc-microbit-counter-using-a-7-segment-display/
https://www.robotique.tech/type/microbit/