



basic education

Department:
Basic Education
REPUBLIC OF SOUTH AFRICA

CURRICULUM AND ASSESSMENT POLICY STATEMENT

(CAPS)

INFORMATION TECHNOLOGY

Section 1

National Curriculum and Assessment Policy Statement for Information Technology

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contemplated in subparagraphs (a) and (b) comprise the following policy documents which will be incrementally repealed by the *National Curriculum Statement Grades R – 12 (January 2012)* during the period 2012-2014: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

- (a) 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.
- (b) 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.

(c) 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.

(d) The National Curriculum Statement Grades R - 12 aims to produce learners that are able to:

- 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.

(e) 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	6	6	7
▪ Beginning Knowledge	(1)	(1)	(2)
• Creative Arts	(2)	(2)	(2)
• Physical Education	(2)	(2)	(2)
• Personal and Social Well-being	(1)	(1)	(1)
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

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

SUBJECT	HOURS
Home Language	6
First Additional Language	5
Mathematics	6
Natural Science and Technology	3,5
Social Sciences	3
Life Skills	4
▪ Creative Arts	(1,5)
▪ Physical Education	(1)
▪ Personal and Social Well-being	(1,5)
TOTAL	27,5

1.4.3 Senior Phase

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

SUBJECT	HOURS
Home Language	5
First Additional Language	4
Mathematics	4,5
Natural Science	3
Social Sciences	3
Technology	2
Economic Management Sciences	2
Life Orientation	2
Arts and Culture	2
TOTAL	27,5

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, <i>National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12</i> , 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.

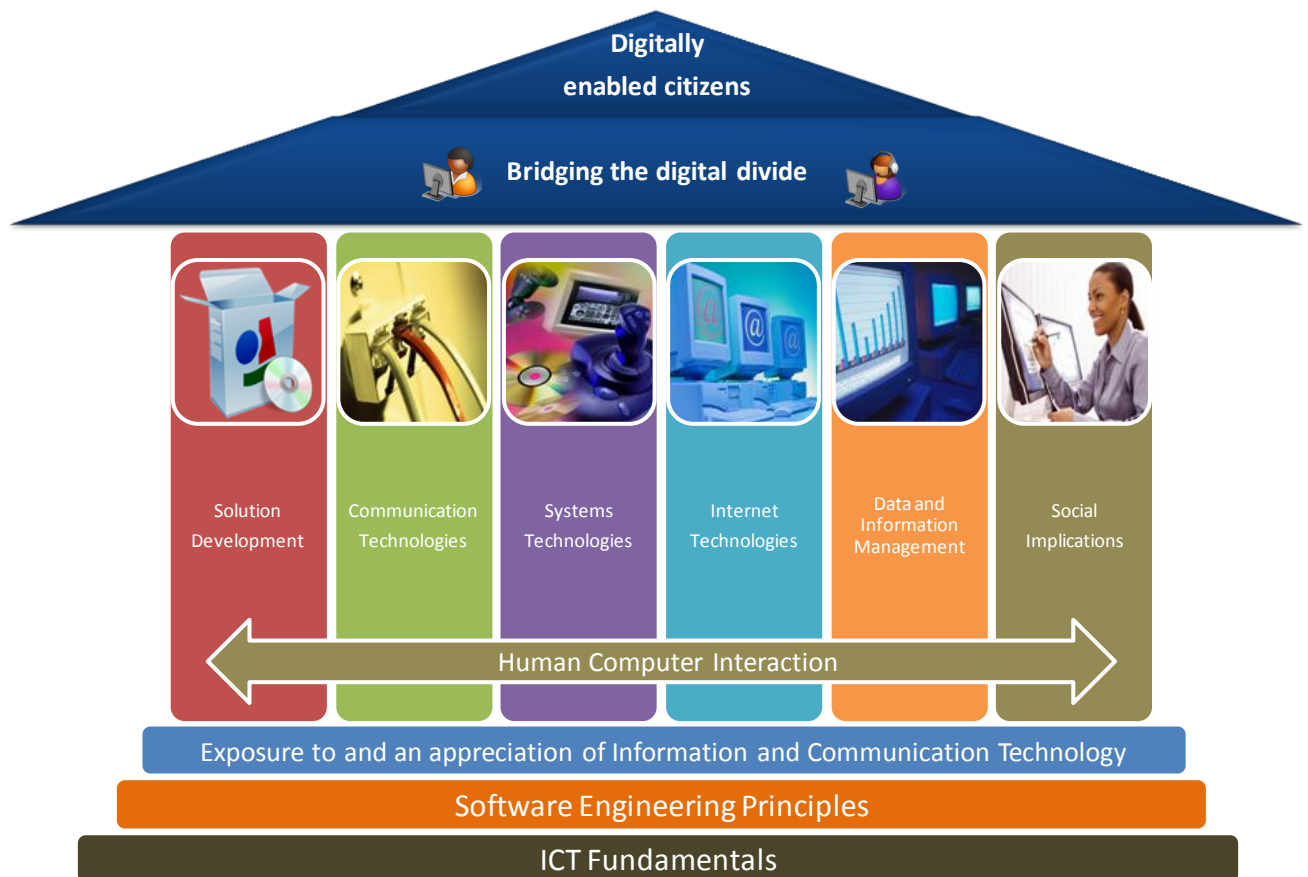
Section 2

Information Technology

2.1 What is Information Technology?

Information Technology is the study of the various interrelated physical and non-physical technologies used for the capturing of data, the processing of data into useful information and the management, presentation and dissemination of data. Information Technology studies the activities that deal with the solution of problems through logical and computational thinking. It includes the physical and non-physical components for the electronic transmission, access, and manipulation of data and information.

The diagram below illustrates how the six main topic areas of the Information Technology curriculum support the teaching of digitally informed learners.



The table below provides the six topics and sub-topics to be covered in Information Technology in grades 10 – 12 and the resources required for teaching IT:

Topic Area	Sub-Topics	Weighting (Content)	Resources
Solution Development	Algorithms and Problem Solving Introduction to Solution Development Application Development Software Engineering Principles	±60%	Computers Textbook Software <ul style="list-style-type: none"> • Introductory graphical programming language • Database Management Software • High-level programming language within a visual development environment using an IDE with a GUI builder • Internet • Browser
Communication Technologies	Networks E-communication	±7%	
Systems Technologies	Introduction to Computers Hardware Software Computer Management	±10%	
Internet Technologies	Internet World Wide Web Internet Services	±8%	
Data and Information Management	Data Representation Database Management Database Design	±10%	
Social Implications	Legal Issues Ethical Issues Social Issues Environmental Issues Health Issues Computers and Society	±5%	

Topic links and overlap

It is important to note that there will always be a degree of overlap between topics. Solution development is enabled by systems technologies in the form of application software. Systems technologies allow for electronic communication. Electronic communication technologies enable the Internet, which is used for various applications that include information dissemination and electronic data interchange. Data and information management is a key concept and secondary activity overlapping concepts in many other areas such as solution development and Internet technologies. Data and information management is enabled by systems technologies. All ICT activities are primarily driven by human involvement, need and intervention, which in turn give rise to social and ethical issues.

For example, when teaching Communication Technologies, one could incorporate the social implications involved. This is also applicable to the Systems Technologies topic where the relevant social implications could be highlighted.

Approach

The curriculum is designed to introduce learners to the breadth of the field of Information Technology.

2.2 Specific aims of Information Technology

In Information Technology a learner will:

- use appropriate techniques and procedures to plan solutions and devise algorithms to solve problems using suitable techniques and tools’
- understand and use appropriate communication technologies for information dissemination;
- appreciate and comprehend the various systems technologies used in the developing of a computer-based system;
- understand that all ICT systems are built upon software engineering principles;
- understand and use Internet technologies for various tasks;
- comprehend and apply the concepts of data and information management to understand how a knowledge-driven society functions; and
- understand the social implications of ICTs and how to use ICT technologies responsibly.

2.3 Time allocation of Information Technology in the curriculum

In Grades 10 and 11 the time allocation for IT is 4 hours per week for 35 weeks. 5 weeks of the school year are taken up by examinations.

The Grade 12 time allocation is 4 hours per week for 28 weeks; 12 weeks of the school year are for examinations.

The table below provides suggestions for the *approximate* teaching time per topic:

	Grade 10		Grade 11		Grade 12	
Topic	Hours	Weeks	Hours	Weeks	Hours	Weeks
Solution Development	92	23	90	22.5	68	17
Communication Technologies	4	1	8	2	4	1
Systems Technologies	16	4	10	2.5	10	2.5
Internet Technologies	14	3.5	6	1.5	4	1
Data and Information Management	8	2	18	4.5	8	2
Social Implications	6	1.5	8	2	6	1.5
Teaching Time: Total	140	35	140	35	100	25
Examinations	20	5	20	5	48	12
TOTAL:	160	40	160	40	148	37

2.4 Resources required for offering Information Technology

Refer to circular S7 of 2006 for details on the resource requirements for the teaching of IT in Grades 10 – 12.

Infrastructure, equipment and finances for the subject are the responsibility of the school.

In Information Technology learners are required to work individually on a computer during contact time and need access to the Internet.

Schools should have a business plan for the subject that addresses the following:

- Initial capital layout for setting up a computer laboratory. The layout should provide for the following:
 - Entry-level computers (to ensure a lifespan of 4 – 5 years), networked
 - One computer per learner per period (during contact time)
 - Provision for sufficient computers to enable the practical examination to be completed in **two sittings**
 - One high-speed printer per computer room
 - Internet access
 - Data projector or demonstrating software
 - Software (operating system, Office suite, security software – antivirus, Internet security, software for solution development)
- Budget
 - Annual running costs
 - Software licensing (operating system, application software, security software, solution development software)
 - Cartridges, paper and storage media
 - Breakage and maintenance (regular service plan)
 - Insurance
 - Internet connectivity
 - Sustainability plan
 - To upgrade or replace software and equipment every 4 – 5 years.

Requirements for high-level programming tool to be used for software development:

High-level software development tool that includes an integrated development environment which:

- supports both structured and object oriented methodologies;
- uses a visual development environment with a graphical user interface builder; and
- allows for event-driven programming.

The GUI builder should allow for component based development with a WYSIWIG (what you see is what you get) editor utilising an event-driven architecture.

The development tool could also include software design utilities to facilitate the application of software engineering practices.

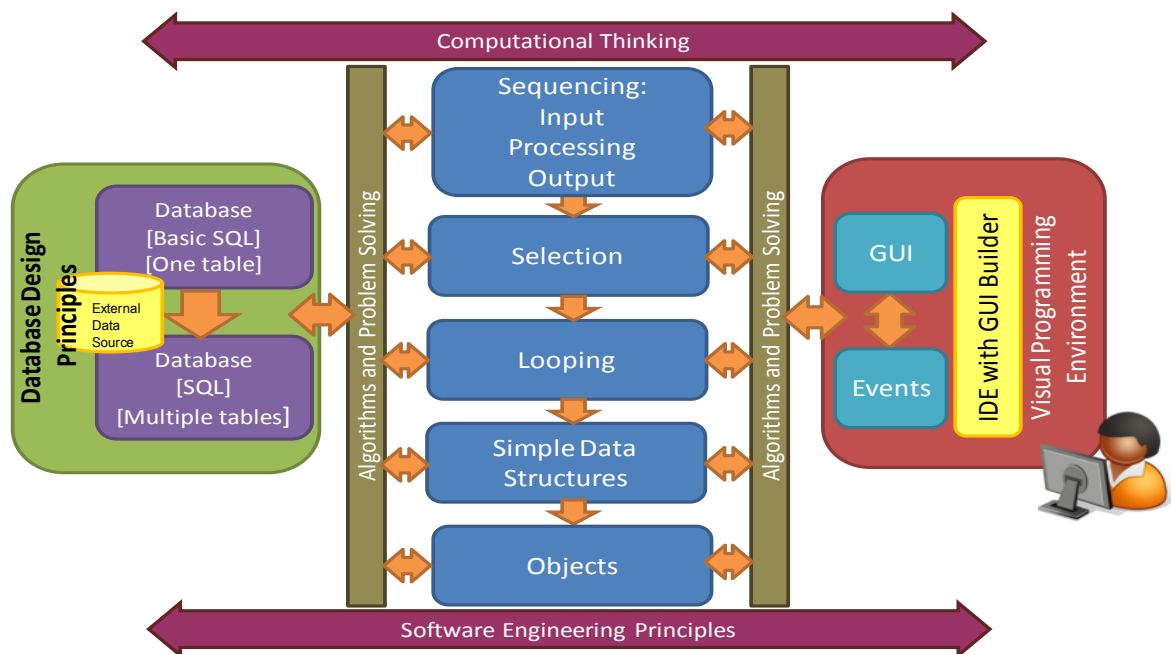
Section 3

Content and scope per topic

3.1 Solution Development

Solution development is the development of software in a planned and structured process and is based on solving computational problems which include data-related problems through logical thinking. It involves the practices of algorithm development and creating a software solution according to a set of rules and/or requirements specified in the problem statement or by a client/business/individual. The software is developed using appropriate problem-solving techniques, tools and methodologies. Software solution development is achieved through computer programming which could be based on a single or combination of development paradigms such as event-driven programming, object-oriented programming and sequential programming.

Broad topic layout and progression



Note:

Basic programming principles and constructs are introduced in Grade 10 through an easy-to-learn, fun tool. An introductory graphical programming teaching tool such as Scratch/BYOB Scratch is used to introduce learners to important computational skills and concepts, algorithm development, problem solving and programming.

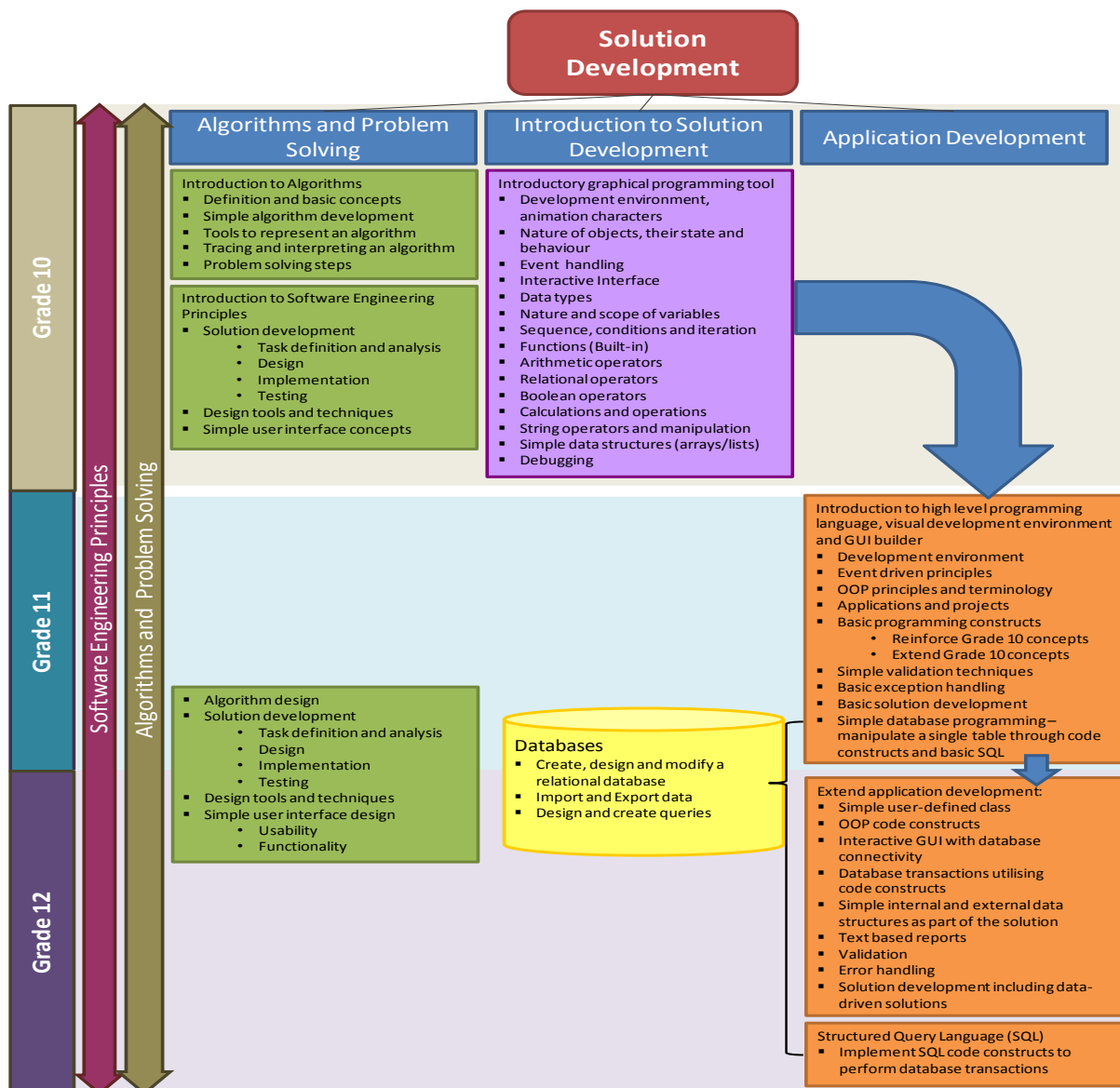
In Grade 11, learners build on the principles and concepts learned in Grade 10 using a high-level programming language that uses an integrated development environment with a GUI builder. Learners are introduced to controls and code and basic object oriented programming (OOP). Event handling principles are reinforced using the form class, attributes, methods and controls.

Skills to manipulate a database through code constructs are also introduced in Grade 11.

In Grade 12, the principles and constructs are further emphasised through more advanced concepts and problems and learners should be ready to engage with basic structured query language (SQL) code and manipulating a relational database.

The development of computational thinking practices of algorithm development, problem solving and programming underpin solution development and should be emphasised from Grade 10 to Grade 12.

Usability, HCI (human computer interaction) and software engineering principles should be reinforced as part of software development as well as when dealing with websites as part of the Internet Technologies topic.



Note:

Algorithmic problem solving in Grade 10 should be dealt with separately at first as an introduction to solution development to develop the learner's computational thinking practices of algorithm development, problem solving and programming using everyday scenarios.

Learners should develop an understanding of the importance of order and precision when developing an algorithm as well as the place of algorithms in software solutions and computing science. Thereafter it should be reinforced, extended and integrated with solution development and programming.

Solution development includes computational thinking and the application of software engineering principles using event-driven programming within the object-oriented (OO) paradigm.

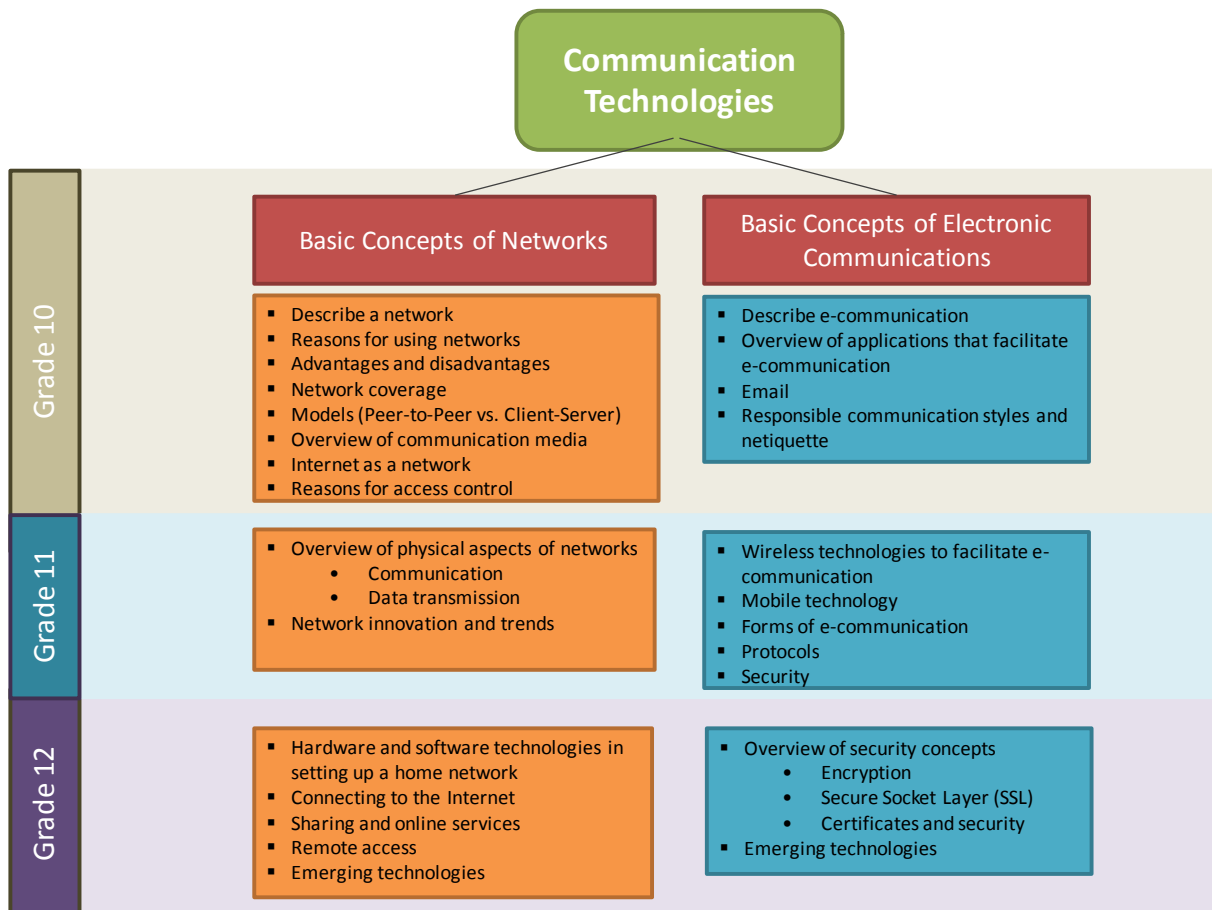
Learners should be able to use appropriate practices and tools to:

- solve computational problems through:
 - identifying and analysing requirements for a specific problem;
 - designing effective algorithms;
 - converting these to code; and
 - testing the solution to see if it meets the requirements.
- apply the principles of human computer interaction to design functional user interfaces.

3.2 Communication Technologies

Communication technologies include various network technologies to facilitate the management and dissemination of digital data from one point to another. Communication technologies also refer to the electronic systems used for electronic data interchange that facilitate, among others, communication and information dissemination between various individuals or groups at a single point or dispersed locations.

Broad topic layout and progression



Note:

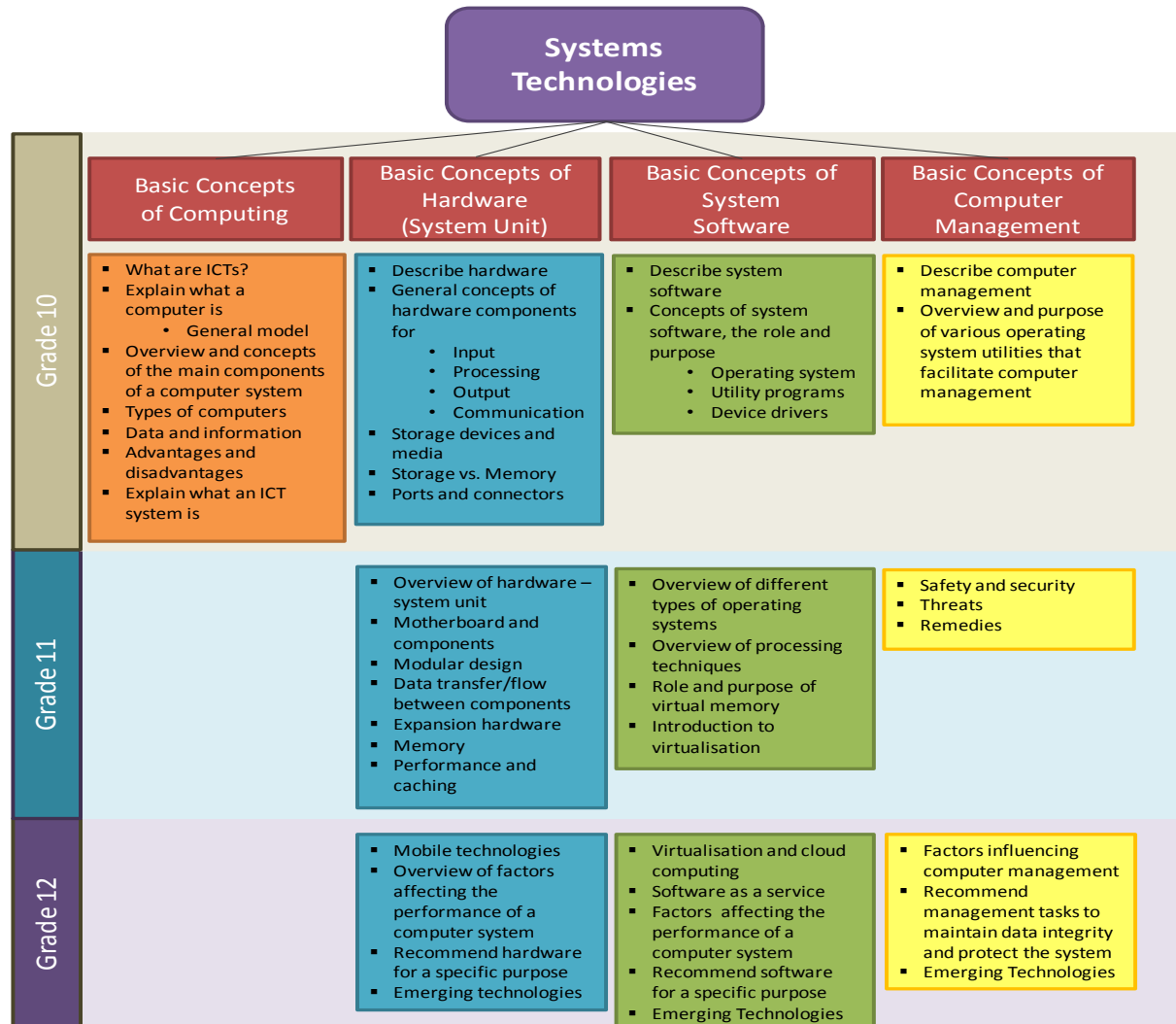
Communication Technologies should teach learners to:

- understand the concepts of the various technologies, standards and protocols involved in the electronic transmission of data via a computer-based network;
- understand the concepts of the technologies and standards implemented to enable electronic communication;
- understand the purpose and uses of communication software;
- understand how communications technology can benefit specific scenarios;
- be aware of and manage security issues; and
- be aware of new trends and developments.

3.3 Systems Technologies

Systems technologies refer to the physical and non-physical components of a computer system. The components of the system are generally related but unconnected in their original form. The connected components which include hardware, peripherals and software components allow the computer to perform the basic functions of a computing system. The basic functions of a computing system include input, processing, output, storage, communication and transfer of data in an electronic format.

Broad topic layout and progression



Note:

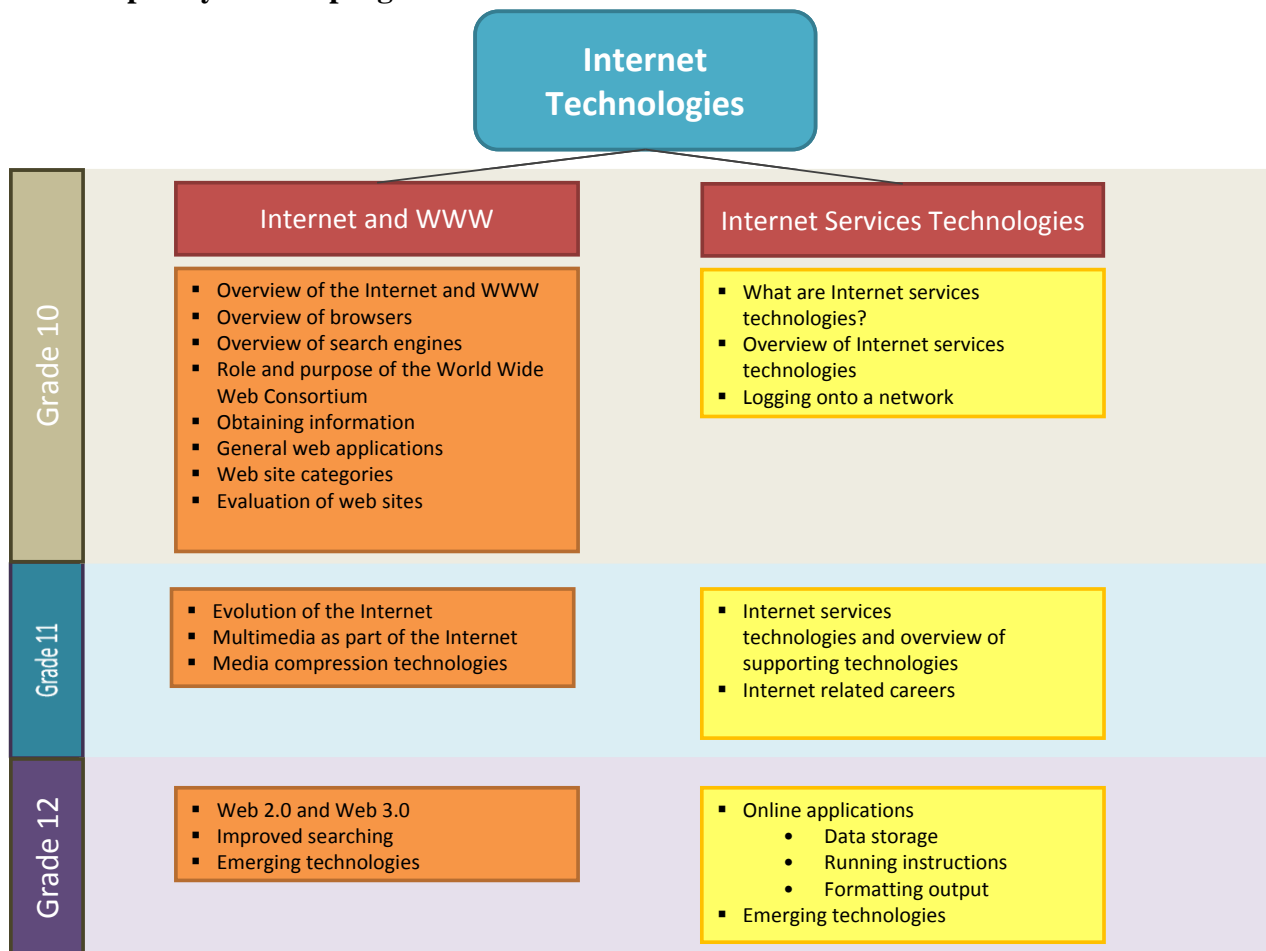
Systems Technologies should teach learners to:

- understand the hardware and software concepts that make up a computer system;
- make informed purchase decisions and whether to upgrade or buy new equipment or software;
- select the most appropriate hardware and software for a given scenario;
- understand how technology can benefit the user in specific contexts;
- understand the operations involved in the management and optimal utilisation of a computer system;
- troubleshoot at an elementary level; and
- be aware of new trends and developments and how to integrate these with existing or new equipment.

3.4 Internet Technologies

Internet Technologies are related and interconnected technologies which enable the establishment of global networks, for various purposes such as collaboration, electronic data interchange, electronic commerce and social networking. Internet services technologies refer to a range of technologies and tools for the design, development and maintenance of websites. The field of Internet services technologies includes Internet programming as well as the roles and responsibilities of each of the individuals involved. Internet technologies include the WWW and all interrelated processes in the digital presentation of multimedia data on a web page.

Broad topic layout and progression



Note:

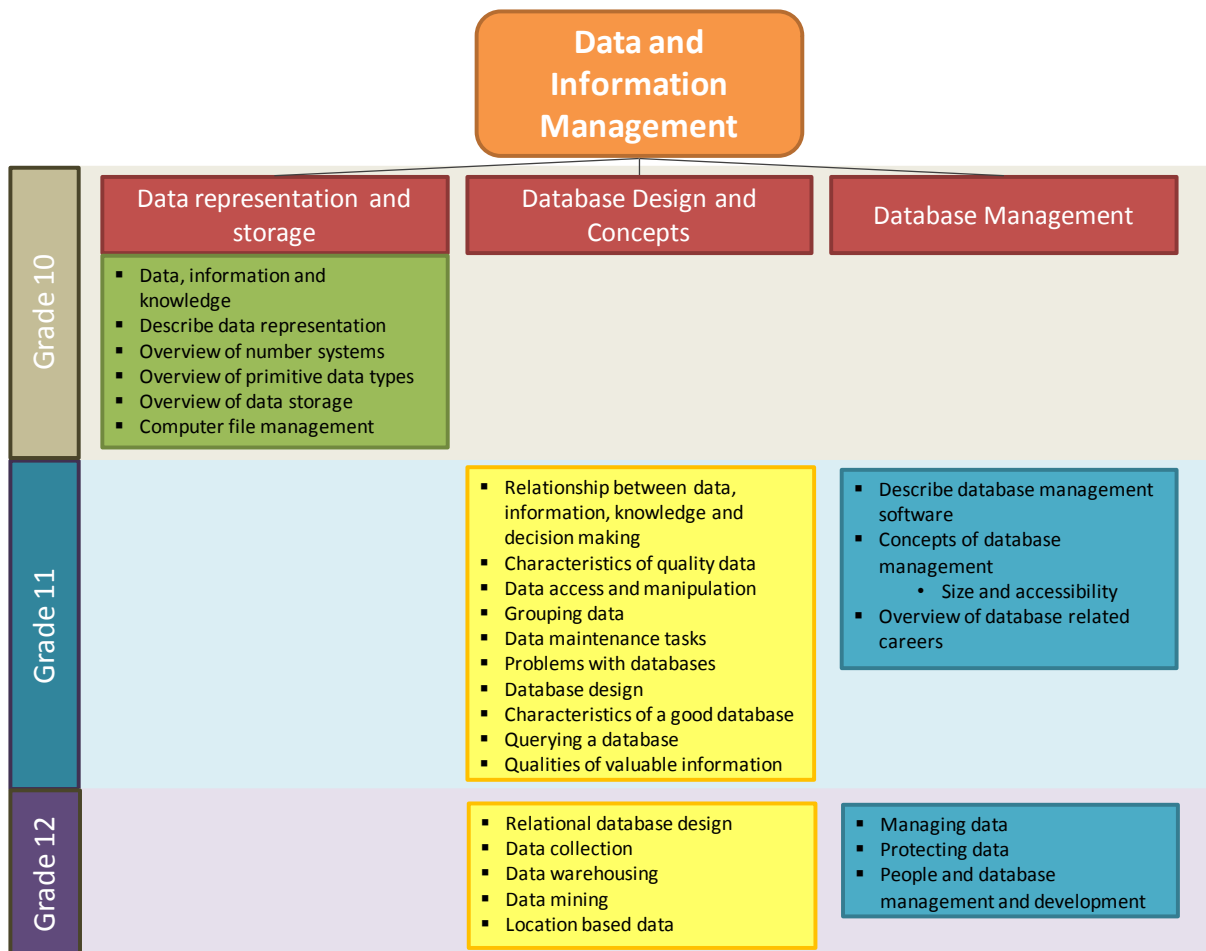
Internet Technologies should teach learners to:

- understand the role that the Internet and the WWW play as part of the global information super-highway and the contribution towards the digital age;
- understand the role of Internet services and supporting technologies;
- understand how Internet technology and services can benefit specific scenarios; and
- be aware of new trends and developments.

3.5 Data and Information Management

Data and information management refers to the techniques and technologies involved in the collection, storage, dissemination and processing of data into information that results in knowledge and leads to decision making. It includes database design principles with specific reference to data storage, retrieval and information presentation design.

Broad topic layout and progression



Note:

Learners need to develop an understanding of:

- data and information with regard to the representation and classification thereof;
- how business takes advantage of computer databases to store data and retrieve information that enables it to gain a competitive edge as well as the social, legal and ethical issues involved;
- database design for use as part of information-driven ICT systems and platforms; and
- DBMS software and its purpose and application in an information-driven society.

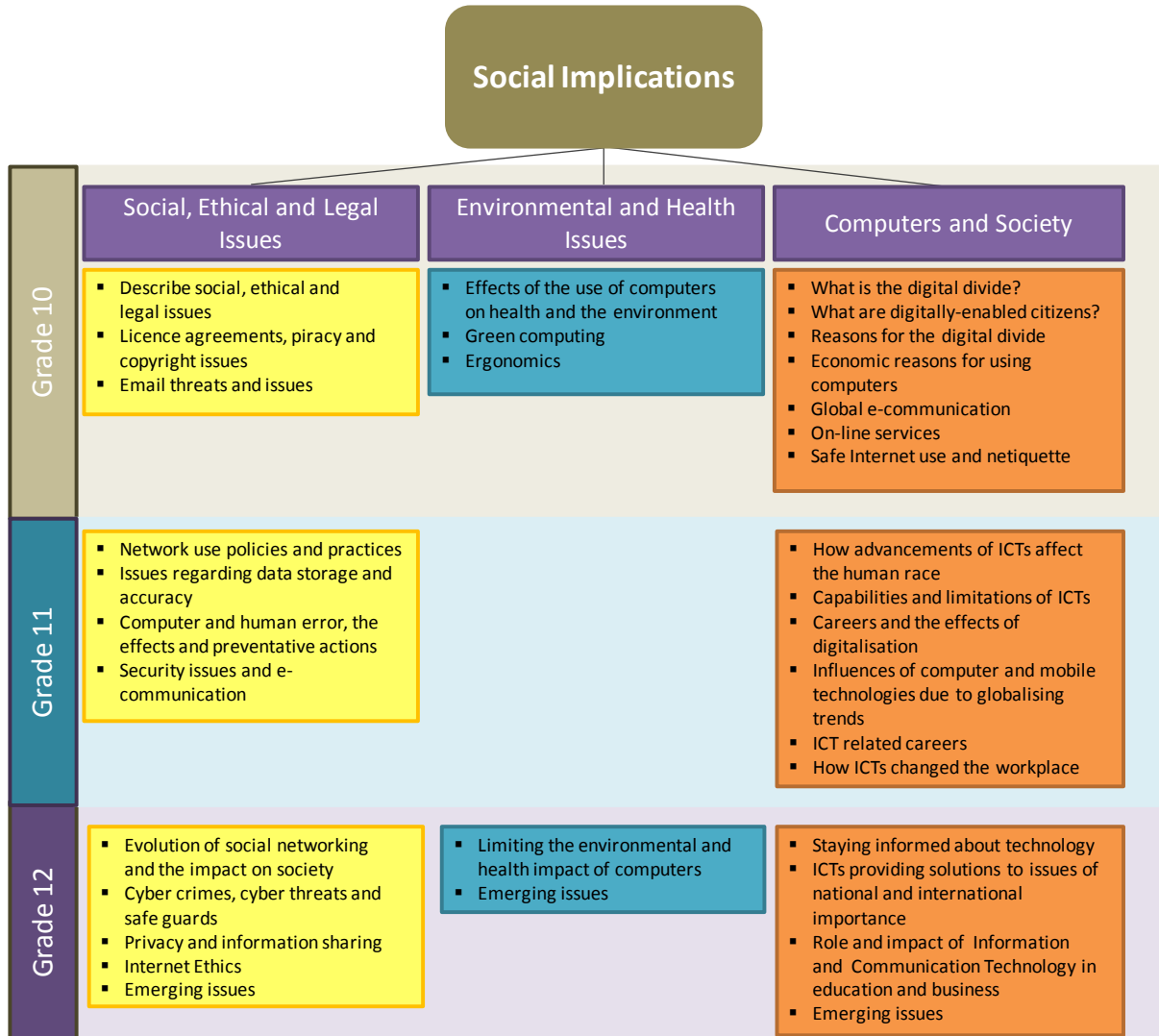
Database design, queries and reports should be linked to application development as described in the topic Solution Development.

This section also covers some practical aspects regarding learning about and working with databases.

3.6 Social Implications

Social implications in the IT curriculum refer to issues relating to the digital age, bridging the digital divide and the responsible use of ICTs.

Broad topic layout and progression



Note:

This topic should provide an overview and understanding of:

- social issues related to the use of computers and how ICTs affect modern life;
- risks and safety aspects that may be involved in the operation of computing equipment within a given context;
- risks and safety issues relevant to using the Internet; and
- principles for making informed decisions regarding the responsible use of ICTs.

Most of the content of Social Implications should be dealt with and integrated with other topics and should not be taught as a stand-alone topic. The time scheduled for this topic can therefore be added to other topics.

3.7 Suggested teaching plan

The suggested teaching plan indicates the minimum content to be covered per term. The sequence of the content or topics listed per term **is not prescribed**. Teachers should design their own work schedules (or use/adapt the work schedule provided in their textbook) to teach the content per term in **appropriate sequence** and pace.

The sub-topics presented in the term plans should not be seen as stand-alone topics. Relevant sub-topics or content should be presented in an integrated manner. Integrating the topics in the lesson presentation should flow naturally due to the nature, links and ‘overlap’ of the content. Some content from one sub-topic may strengthen and underpin the content of another. This approach should be applied throughout the three-year curriculum.

It is important that the specific technologies in the teaching plans are revised at regular intervals to phase out old technologies and to include new technologies.

As the length of terms varies from one year to another, the teaching plan/work schedules should be adapted accordingly on a year-to-year basis.

3.7.1 Grade 10

Grade 10: Term 1 – 10 weeks/40 hours
Systems Technologies: Basic concepts of computing (± 1 week/4 hours)
<ul style="list-style-type: none">• What are Information and Communication Technologies (ICTs)?• Define Information Technology• Explain what a computer is: Overview of a general model of a computer in terms of input, storage, processing, output and communication• Overview and concepts of the main components of a computer system:<ul style="list-style-type: none">▪ Hardware vs software▪ Common/generic physical components of a home computer system: input (keyboard, mouse), storage (hard drive), processing (CPU and RAM), output (monitor, printer) and communication (modem/router)▪ Common/generic non-physical components of a home computer system: system software (operating system) and application software<ul style="list-style-type: none">○ Generic/common examples and uses○ What are shareware, freeware, free open source software (FOSS), proprietary software?▪ Concept of interdependency of hardware and software• Types of computers: desktop, notebooks, netbooks, tablets, smart phones, server, embedded computers (microcontrollers): purpose and uses<ul style="list-style-type: none">▪ Differentiate between the types of computers in terms of primary uses, processing power and size▪ Categorise computers/classification of computers in terms of portability/mobility, processing power and usage• Advantages and disadvantages of using computers• Explanation of and differentiation between data and information:<ul style="list-style-type: none">▪ Information processing cycle: input, processing, output, storage, communication▪ Transition from raw data to processed/organised information▪ Overview of uses and examples of information within an organisation▪ Why is information useful?• What is an ICT system?<ul style="list-style-type: none">▪ Overview of a general model of an ICT system: convey, manipulate and store data▪ Example of an ICT system (familiar context, e.g. point-of-sales system, cell phones)
Data and Information Management: Data representation and storage (±2 weeks/8 hours)
<ul style="list-style-type: none">• Data, information and knowledge• What is data representation?• What is data storage?• Bits and bytes• Overview of number systems: decimal, binary, hexadecimal<ul style="list-style-type: none">▪ Conversion between:<ul style="list-style-type: none">○ binary and decimal and vice versa○ decimal and hexadecimal and vice versa▪ Overview of digital character representation, e.g. ASCII/UTF-8, Unicode• Overview of primitive data types and their storage (integer types, text/string types, floating point types)• Overview of data structures and collections of data: storage in terms of:<ul style="list-style-type: none">▪ Files, databases▪ Reasons for data storage• Computer file management:<ul style="list-style-type: none">▪ Organising files▪ Files, folders and drives▪ File specification: drive, path, filename, file extension▪ File manager▪ Hierarchical structure▪ Reasons for having a file structure▪ Manipulating files and folders▪ File-naming conventions▪ Common file types and extensions (association)<ul style="list-style-type: none">○ Archived and compressed○ Forms of text files○ Database, spreadsheet, presentations and word processing documents○ Graphic files, movie, sound and animation files○ Font files○ Source code○ Object code, executable files, shared and dynamically linked libraries▪ Saving as another type/version and exporting between file types

Grade 10: Term 1 – 10 weeks/40 hours	
Social Implications ($\pm 1/2$ week/2 hours)	
<ul style="list-style-type: none"> • Social issues applicable to term 1 content such as licence agreements (including creative commons), piracy, copyright, copyleft • What are social, ethical and legal issues pertaining to ICTs? • Economic reasons for using computers: saving paper, labour, communication costs, efficiency, accuracy and reliability • Digital divide <ul style="list-style-type: none"> ▪ What is the digital divide? ▪ What are digitally enabled citizens? ▪ Reasons for the digital divide 	
Solution Development: Introduction to Algorithms (± 2 weeks/8 hours)	Notes
<ul style="list-style-type: none"> • Basic concepts of an algorithm <ul style="list-style-type: none"> ▪ What is an algorithm? Develop a clear understanding • Examples of algorithms in everyday life, e.g. instructions to draw a kite or fold a paper jet, recipe to bake a cake <ul style="list-style-type: none"> ▪ Devise an algorithm/basic instructions to complete similar tasks ▪ Use a tool, e.g. basic flowchart to describe a task ▪ Interpret a basic flow chart • Explore algorithms such as: <ul style="list-style-type: none"> ▪ Determine smallest, largest value of more than two values ▪ Swapping values ▪ Determining aggregates, e.g. sum ▪ Basic calculations such as calculating area, volume, VAT ▪ Determine whether a number is even ▪ Determine whether a number is a factor of another number • Produce an algorithm to solve a problem • Tools, e.g. basic flow charts/pseudo code to represent an algorithm • Trace an algorithm to determine the outcome– trace table • Compare algorithms considering, e.g. order, precision and efficiency • Value of accurate, well-tested algorithms 	<p>The purpose of this section is to serve as an introduction to solution development to develop the learner’s computational thinking practices of algorithm development, problem solving and programming using everyday scenarios.</p> <p>Exploring algorithms to solve generic problems will enable a learner to use similar principles to devise algorithms for new problems or situations. It will also enable the learner to identify the types of problems requiring certain generic algorithms.</p> <p>Investigating specific algorithms should provide the learner with the opportunity to explore various ways to solve the same problem by using different principles or tools.</p>
Solution Development: Introduction to solution development using an Introductory Graphical Programming Tool ($\pm 4 1/2$ weeks/ 18 hours)	Notes
<ul style="list-style-type: none"> • Introduction to the programming tool, basic terms and development environment • Short animated sequence/cartoon strip/movie • Exploring the use of variables <ul style="list-style-type: none"> ▪ Global variables vs local variables • Variable naming conventions • Assigning values to variables • Exploring data types: integers, strings, floats, Boolean • Keyboard input, mouse input • Operators (+ , , * , /) and order of precedence • Retrieving remainders: modulus • Comparison operators and performing logical comparisons • Functions – random, round, square root • Basic calculations such as area, volume, VAT and simple formulae, typical calculations done in other subjects • Conditional constructs (if and if-then-else) • Applying algorithms such as swapping values, finding aggregates, isolate digits in an integer number, finding the smallest/biggest of two numbers, determine if a number is a factor of another number, determine if a number is even • Event handling (When clicked, When key pressed, Broadcast and When I receive) <ul style="list-style-type: none"> ▪ Sensing events/actions and responding programmatically 	<p>The purpose of this section is to introduce programming environments and associated terminology using animated characters as objects and to explore the nature of objects, i.e. their state and behaviour. Through properties, methods (scripts) and events (triggers), explore “event-driven programming” and the role of messaging in OOP.</p>
Formal Assessment (PoA):	Reporting
1 practical test + 1 theory test covering content taught to that point	Add raw marks and totals of the 2 tests and convert to percentage to determine term mark

Grade 10: Term 2 – 10 weeks/40 hours, including examination (2 weeks)
Systems Technologies: Basic concepts of hardware (± 1 week/4 hours)
<ul style="list-style-type: none"> • Describe hardware • Extend hardware concepts <ul style="list-style-type: none"> ▪ Input devices: purpose and uses (What is it? Where is it used? What is it used for?) <ul style="list-style-type: none"> ○ Alternative keyboards, pointing devices, touch screens, touch-sensitive pads, pen input, game controllers, digital cameras, video input, scanners, reading devices, data collection devices, biometric input, toy/electronic device interfaces ○ Transfer/synchronise between computer and phone ▪ Output devices: purpose and uses (What is it? Where is it used? What is it used for?) <ul style="list-style-type: none"> ○ Display devices, printers, data projectors, interactive whiteboards, toy/electronic device interfaces ○ Concepts regarding quality of output and speed where applicable ▪ Storage devices: purpose and uses (What is it? Where is it used? What is it used for?) <ul style="list-style-type: none"> ○ Hard drives (fixed and portable), USB flash drives, U3 smart drive, solid state drives, memory cards, optical disks, DVD and Blu-ray drives and media ○ Capacity, portability, use ▪ System unit (processing: CPU and RAM) <ul style="list-style-type: none"> ○ General function of CPU and RAM ▪ Communication devices (modem/router) <ul style="list-style-type: none"> ○ General function ▪ Identify ports and connectors and their purpose: USB, Firewire ▪ Categorise hardware according to input, output, storage, processing and communication devices • Memory vs storage • Compare input, processing, output, storage and communication devices of a desktop computer with a smart phone <ul style="list-style-type: none"> ▪ Which are the same? Which are different? Why are they the same/different?
Systems Technologies: Basic concepts of system software (± 1 week/4 hours)
<ul style="list-style-type: none"> • Describe system software • Extend system software concepts <ul style="list-style-type: none"> ▪ Operating system <ul style="list-style-type: none"> ○ What is an operating system? ○ What is the purpose/role of an operating system? <ul style="list-style-type: none"> ◆ General role: suite/group of related programs which manage hardware and software ◆ Specific role: provides user interface, I/O management ◆ Brief overview of the role of the operating system in terms of file, disk, memory, storage and process management ○ Types of operating systems (also associate with types of computers), e.g. stand-alone, network, embedded ○ Examples of common operating systems ▪ Utility programs <ul style="list-style-type: none"> ○ What are utility programs? ○ What are they used for? ○ Generic/common examples ▪ Purpose of device drivers
Social Implications (±½ week/2 hours)
<ul style="list-style-type: none"> • Social issues applicable to term 2 content such as ergonomics, green computing issues, health issues • Global e-communication, i.e. accuracy, time, distance, communication costs, speed

Solution Development: Software Engineering Principles ($\pm 1\frac{1}{2}$ week/6 hours)	Notes
<ul style="list-style-type: none"> • What is problem solving? • Problem solving steps (<i>Polya, G., 1957</i>) <ul style="list-style-type: none"> ▪ Understand the problem (task/problem description or scenario/user stories) <ul style="list-style-type: none"> ○ State in own words ○ Clarity on what needs to be done ○ What is known or given? What is missing or needed? ▪ Devise a plan/algorithm (storyboard – visual or textual) <ul style="list-style-type: none"> ○ Look for patterns ○ Look at related problems, known solutions ○ Examine simpler or special cases ○ Make a table, create diagram, use guess and check, work backwards, identify sub-goal ▪ Carry out the plan/implement the algorithm (write the code) ▪ Look back/test (see if it works) <ul style="list-style-type: none"> ○ Check results against original problem. Does it make sense? Is there another solution? • Solve a problem using the problem solving steps • Use appropriate tools and techniques used in software analysis, viz.: <ul style="list-style-type: none"> ▪ User stories (written by the client and provide the requirements) ▪ Noun-verb analysis of user stories <ul style="list-style-type: none"> ○ List of nouns provides identification of objects and state ○ List of verbs provides identification of behaviour ▪ Acceptance tests (does the program meet the requirements?) 	<p>The purpose of this section is to teach problem-solving procedures and techniques.</p> <ul style="list-style-type: none"> • Scenario/Story (description of the story or problem/task statement) • Design storyboard – visual (state-transition-diagram), textual (algorithm) or flow chart (how to go about creating the animation/solving the problem) • Convert to programming code (write the program) • Test (see if it works)
Solution Development: Introduction to solution development using an Introductory Graphical Programming Tool (± 4 weeks/ 16 hours)	Notes
<ul style="list-style-type: none"> • Extend the use of variables, logical operators, random numbers and built-in functions • Boolean logic/operators (and, or, not) • Conditional constructs (if and if-then-else) including Boolean operators • Strings <ul style="list-style-type: none"> ▪ String operators such as concatenate/combine strings ▪ String operations such as comparing strings • Interactive user interface (objects, e.g. animated character, as buttons) • Iteration constructs (for), pre-conditional and post-conditional (repeat, repeat until and forever) • Basic validation techniques (input and processing), e.g. test for negative number when calculating square root • Debugging techniques • Debugging using the variable watch facility 	<p>Implement algorithms to solve general computing problems in similar categories to those given below:</p> <ul style="list-style-type: none"> • Finding the smallest/biggest of more than two numbers • Determine whether a number is a prime number • Lowest common multiple (LCM), greatest common divisor (GCD) • Find a specified character in a string • Simple diagramming and animation • Simple condition-based drawing
Assessment (PoA):	Reporting
1 test + 1 examination (1 practical paper + 1 theory paper)	Add raw marks and totals of the test and two papers and convert to percentage to determine term mark

Grade 10: Term 3 – 10 weeks/40 hours
Communication Technologies: Networks (±½ week/4 hours)
<ul style="list-style-type: none"> • Describe a network • Reasons for using networks such as communication, access to/sharing resources, centralisation, file and funds transfer, productivity, leisure • Advantages and disadvantages of networks • Overview of different communication media (wired vs wireless) <ul style="list-style-type: none"> ▪ Types of cabling and components ▪ Types of transmitters and components • Local area network (LAN) vs. wide area network (WAN) – coverage and where it is used • Internet as an example of a network (WAN) • Differentiate between client-server and peer-to-peer networks • Explain the reasons for logging into a network and connecting to a server – access control
Communication Technologies: Electronic Communications (±½ week/2 hours)
<ul style="list-style-type: none"> • Describe electronic communication • Overview of applications/tools to facilitate e-communication – purpose and uses (What is it? What is it used for?) <ul style="list-style-type: none"> ▪ E-mail, web browser, File Transfer Protocol (FTP), instant messaging, chat rooms, video conferencing and Voice over Internet Protocol (VoIP), RSS aggregator, weblog, text, picture and video messaging – examples • E-mail as a form of e-communication <ul style="list-style-type: none"> ▪ Uses of e-mail ▪ E-mail accounts (Internet Service Provider (ISP) and web-based) ▪ E-mail addresses ▪ Use e-mail • Responsible communication styles and netiquette
Systems Technologies: Computer Management (±1 week/4 hours)
<ul style="list-style-type: none"> • Describe computer management • Overview and purpose of various management tasks and operating system utilities <ul style="list-style-type: none"> ▪ Management of desktop ▪ Management of files and folders ▪ General housekeeping tasks ▪ Defragmentation ▪ Scheduling/updating ▪ Archive, backup ▪ Compress/decompress files ▪ Security features – firewall, anti-virus, control of spyware, adware ▪ Installing/uninstalling software (custom and full installation, product keys, activation codes) ▪ Add devices/drivers ▪ System settings and properties
Internet Technologies: Internet and WWW (±1½ week /± 6 hours)
<ul style="list-style-type: none"> • Overview of the Internet <ul style="list-style-type: none"> ▪ Describe the Internet ▪ Internet addresses – Internet protocol (IP) addresses and domain names • What is needed to connect to the Internet referring to <ul style="list-style-type: none"> ▪ Internet Service Providers (ISPs), wired and wireless connections • Overview of the World Wide Web (WWW) <ul style="list-style-type: none"> ▪ Describe the WWW ▪ Web address/uniform resource locator (URL) ▪ Web page and website ▪ Types of websites, their purpose/what they offer and examples <ul style="list-style-type: none"> ○ Portal, news, informational, business, weblog (blog), Wiki, online social network, educational, entertainment, advocacy, web application, content aggregator, personal • Criteria to evaluate websites <ul style="list-style-type: none"> ▪ Affiliation (e.g. who supports the website?) ▪ Audience (e.g. level at which it is written/who is it intended for?) ▪ Authority (e.g. who is the author and what are his/her credentials?) ▪ Content (e.g. organisation of content and working links) ▪ Currency (e.g. is the information on the web page up-to-date?) ▪ Design (e.g. is it easy to navigate and visually pleasing? How quickly does it download?) ▪ Objectivity (e.g. does it reflect any preconceptions?)

Grade 10: Term 3 – 10 weeks/40 hours	
<ul style="list-style-type: none"> • Browsing and searching <ul style="list-style-type: none"> ▪ Examples of web browsers ▪ What is a search engine? ▪ Examples of search engines ▪ Performing searches using a search engine (search techniques) ▪ How to access and browse a website • What is the World Wide Web consortium (W3C)? 	
Social Implications (±1½ week/2 hours)	
<ul style="list-style-type: none"> • Social issues applicable to term 3 content such as: <ul style="list-style-type: none"> ▪ E-mail threats and issues: viruses, hoaxes, spam, phishing, e-mail spoofing and pharming ▪ Safe email and Internet use: dangers and tips to ensure safe use 	
Solution Development: Introduction to solution development using an Introductory Graphical Programming Tool (±4 weeks/ 16 hours)	Notes
<p>Using good programming principles and algorithmic development extend the use of the tool:</p> <ul style="list-style-type: none"> • Explore lists/arrays concepts (storing and accessing a list of numbers and strings) and containers <ul style="list-style-type: none"> ▪ Manipulating lists such as adding, deleting, replacing, inserting items ▪ Parallel lists • Explore simple nested loops • Develop an elementary game or other suitable programs that exercise the content of the syllabus • Develop simple applications incorporating a combination of graphics, iteration, conditional constructs, concepts covered 	<p>Implement basic algorithms to solve general computing problems using methods such as:</p> <ul style="list-style-type: none"> • Finding a given item in a list <p>Exploring algorithms such as to convert binary numbers, digital character representation</p> <p>Develop and use algorithms to solve various problems</p>
Solution Development: Software Engineering Principles and Practical Assessment Task (PAT) (±2 weeks/8 hours)	Notes
<p>Start with PAT and reinforce software engineering principles, problem-solving techniques and algorithms as well as debugging techniques</p>	<p>Design and develop a solution for a specific problem, including good principles of algorithmic development and problem solving using:</p> <ul style="list-style-type: none"> • Scenario/story (description of the story or problem/task statement) • Design storyboard – visual [state-transition-diagram], textual [algorithm] or flow chart (how to go about creating the animation/solving the problem) • Convert to programming code (write the program) • Test (see if it works)
Assessment (PoA):	Reporting
1 practical test + 1 theory test	Add raw marks and totals of the two tests and convert to percentage to determine term mark

Grade 10: Term 4 – 10 weeks/40 hours, including examination (3 weeks)

Internet Technologies: Internet and WWW (±½ week/4 hours)

- Overview of plug-in applications
 - Describe plug-in applications
 - Examples and purpose of plug-in applications for browsers such as PDF converters and tools, Flash player, Java, QuickTime player, RealPlayer, Silverlight

Internet Technologies: Internet Services Technologies (±1½ week/6 hours)

- What are Internet services technologies?
- Usability of web pages/sites
 - Compare usability issues such as readability, navigation, consistency, layout, typography
 - How does this relate to user interface design?
- Overview of web development
 - Concept of a web page as a file that contains text and HTML and/or XHTML code
 - Tags and elements of a web page
 - Web design principles
- Exploring a simple HTML editor with basic HTML code to view tags and elements
- Utilising an HTML preview tool to explore some of the functionality and purpose of the various tags:

Notes

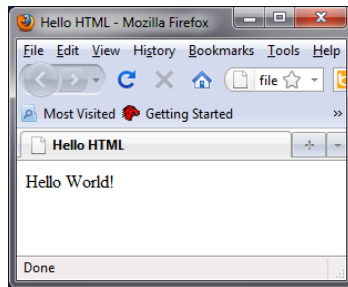
Exposing learners to web development concepts provides practical application of the Internet sections. It also reinforces the idea that computers need explicit commands in a particular format.

It is a good way to demonstrate that what you see is not always what you get.

It could also provide an introduction to the idea of syntax before the high-level programming language is introduced.

```
<html>
<head>
<title>Hello HTML</title>
</head>
<body>
<p>Hello World!</p>
</body>
</html>
```

Display:



As an optional extension, learners could create simple web pages using HTML/XHTML and a text editor such as Notepad using the following:

Basic Document Tags	<pre><html> <head> <title> <body></pre>	<pre></html> </head> </title> </body></pre>
Heading Elements	<pre><h1> <h2> <h3></pre>	<pre></h1> </h2> </h3></pre>
Text Elements	<pre><p>
 <i></pre>	<pre></p> </i></pre>
Image	<pre></pre>	
Ordinary Links	<pre>Link-text </pre>	

Solution Development: Introduction to solution development (±3 weeks/12 hours)	Notes
<p>Using good programming principles and algorithmic development extend the use of the tool:</p> <ul style="list-style-type: none"> • Revise, consolidate and extend solution development content by developing applications incorporating a combination of features <p>Optional extensions (if time permits):</p> <ul style="list-style-type: none"> • Explore new features in new versions such as tools that introduce various strategies for transferring to high-level programming language <p>or</p> <p>Introduce learners to the high-level programming language</p> <p>or</p> <ul style="list-style-type: none"> • Explore external input using specialised hardware such as a sensor board, e.g. to emulate a real-world object or explore robotics using a robotics kit (this extension has cost implications as it requires additional equipment/kits to be purchased) 	<p>Solve various computational problems through identifying and analysing requirements for a specific problem; designing effective algorithms; converting these to code and testing the solution to see if it meets the requirements. In doing so, learners should make use of appropriate practices and tools and consider HCI principles in designing user interfaces.</p>
Solution Development: Software Engineering Principles and Practical Assessment Task (PAT) (±2 weeks/8 hours)	
<ul style="list-style-type: none"> • Finalise PAT <ul style="list-style-type: none"> ▪ Construct a solution based on the planning ▪ Document the solution by adding comments 	
Assessment (PoA):	Reporting (promotion mark)
1 examination (practical paper + theory paper) + PAT	Convert: PAT to 25% Paper 1 to 25% Paper 2 to 25% Term 1 + Term 2 + Term 3 marks to 25%

Grade 11

Grade 11: Term 1 – 10 weeks/40 hours
Systems Technologies: Hardware (±1 week/4 hours)
Extend hardware concepts from Grade 10: <ul style="list-style-type: none">• Overview of hardware as part of the system unit• Describe the motherboard• Purpose and role of the motherboard• Components as part of the motherboard<ul style="list-style-type: none">▪ Purpose and role of a BIOS chip, CPU, RAM, ROM, slots, cards and buses• Modular design• Flow/transfer of data between components<ul style="list-style-type: none">▪ USB – PnP, U3▪ Point-to-point connections▪ Purpose and role of cache memory and caching• Purpose and role of the expansion cards• Memory as part of a computer system<ul style="list-style-type: none">▪ ROM, RAM – role and characteristics▪ Temporary/permanent/magnetic/optic/solid state• Difference in performance of different components and caching (including web caching and disk caching)
Systems Technologies: Software (±1 week/4 hours)
Extend functions of system software from Grade 10: <ul style="list-style-type: none">• Overview of various types of operating systems in terms of cost, size, hardware needed and platform<ul style="list-style-type: none">▪ Programming language compilers/interpreters<ul style="list-style-type: none">○ What are programming language compilers/interpreters?• Overview of processing techniques (managed by systems software)<ul style="list-style-type: none">▪ Multi-tasking, multi-threading, multi-processing• What is virtual memory? Role and purpose• Introduction to virtualisation – overview<ul style="list-style-type: none">▪ Describe virtualisation▪ Virtual machines – purpose and examples
Communication Technologies: Networks (±1 week/4 hours)
<ul style="list-style-type: none">• Overview of physical aspects of a network<ul style="list-style-type: none">▪ Communication (NIC, Wi-Fi, WiMAX, 3G)▪ Data transmission<ul style="list-style-type: none">○ Media (reinforce from Grade 10)○ Physical layout (topology – star)○ Physical limitations (bandwidth)○ Connection (router/bridge)○ Size (PAN/HAN, LAN, WAN)• Overview of network innovation<ul style="list-style-type: none">▪ VoIP▪ Virtual Private Networks (VPN)▪ Location-based computing (3G, wireless, GPS)
Social implications (±½ week/2 hours)
<ul style="list-style-type: none">• Social issues applicable to term 1 content such as network use policies and practices• How the advancement of ICT affects the human race<ul style="list-style-type: none">▪ Computers providing solutions to issues of national and international importance such as<ul style="list-style-type: none">○ Weather, elections, census• Capabilities and limitations of ICTs

Solution Development: Application Development using a high-level programming language ($\pm 1\frac{1}{2}$ week/4 hrs)	Notes
<p>Introduction to the high-level programming language, integrated visual development environment and GUI builder</p> <ul style="list-style-type: none"> • Introduce the programming environment • Integrated development environment (IDE) and event-driven programming • Controls/components • Create, retrieve and save an application/project for development and modification • Controls/components to be used in this section: <ul style="list-style-type: none"> ▪ Form, button, label, panel, radio group, text box, combo box, check box, message box, image and timer <p>Application development</p> <p>Introduction to OOP – basic principles of classes and objects:</p> <ul style="list-style-type: none"> • Introduction to classes and objects with reference to controls/components <ul style="list-style-type: none"> ▪ Form class – the base class ▪ Objects and attributes ▪ Common properties of components ▪ Events, methods (code to handle events) ▪ Terminology: instances, instantiation ▪ Concept of events “sharing” class attributes ▪ Concept of the accessibility/visibility of attributes shared among class methods • Adding controls to a form and enter simple code (simple applications using basic controls that accept input and give output such as simple messages) <ul style="list-style-type: none"> ▪ Manipulate and access simple component properties ▪ Write simple code to display a message, e.g. to display simple messages using a message box ▪ Illustrate the concept of a message parameter (the idea of passing a message to an object) ▪ Add controls and methods to perform tasks such as closing a form, changing colours, administer a login, manipulating the appearance and behaviour of other controls • Simple Graphical User Interface(GUI) design concepts, e.g. functionality and usability issues <ul style="list-style-type: none"> ▪ GUI model, e.g. paper prototype • Naming conventions – controls/components 	<p>Learners will transfer their knowledge and skills with regard to programming principles, concepts and constructs learned and used in Grade 10.</p> <p>Event-driven programming should be taught within the OOP paradigm.</p> <p>This section introduces the basics of objects and classes such as:</p> <ul style="list-style-type: none"> • Properties • Events • Methods (code to handle events) • Naming objects <p>The controls/components are used to introduce basic OOP concepts.</p> <p>The principles of OOP as well as OOP terminology and concepts should be used throughout this section.</p> <p>Although the controls/components are used to introduce OOP concepts, the primary focus of the GUI builder should not be on the controls/components or the number of controls/components used, but rather the construction of a simple graphical interface to illustrate the concepts and terminology and solve a problem.</p>
Solution Development: Application Development using a high-level programming language (± 5 weeks/12 hrs)	Notes
<p>Introduction to computational problems using the high-level language</p> <p>Variables and data types</p> <ul style="list-style-type: none"> • Simple data types (integer, real/decimal, character, string, Boolean) • Variables and scope of variables • Variables and objects as form class attributes • Assignment of values to variables <p>Basic computing</p> <ul style="list-style-type: none"> • Operators: arithmetic, relational and Boolean, including precedence • Integer division and retrieving remainders – modulus • Extend applications by adding auxiliary methods to perform simple calculations in the form class, e.g. volume, area, VAT • Casting from one type to another (integer/real/float/ string/text) • Reinforce and use of simple built-in methods dealt with through the introductory tool, e.g. square root, randomising and rounding • Format number and text output • Concept of a constructor for the initialisation of attributes • Dynamically instantiate simple objects from existing classes which supports the development of the solution, e.g. a list, image and relevant controls by invoking a constructor • Bind an applicable event to the dynamically instantiated object if 	<p>This section reinforces the content and concepts taught in Grade 10 using the high-level programming language and visual environment.</p> <p>Solution development includes computational thinking and application of software engineering principles using event-driven programming within the OOP paradigm.</p> <p>As an introduction, also provide working programs that learners could load and modify to meet new specifications as a way of exploring the new environment and programming language and for learning OOP (remixing).</p>

<p>required as part of the solution</p> <p>Simple decision making constructs</p> <ul style="list-style-type: none"> • Simple decision making and comparison (if and if-then-else) <p>Debugging</p> <ul style="list-style-type: none"> • Errors and debugging techniques <ul style="list-style-type: none"> ▪ Implement watches and traces to identify logical errors in code constructs (compiler-based and paper-based) <p>Validation</p> <ul style="list-style-type: none"> • Basic input and processing validation techniques, e.g. test for division by zero <p>Built-in methods</p> <ul style="list-style-type: none"> • Extend the use of built-in methods and the concept of parameters/message passing <p>Algorithms and problem solving</p> <ul style="list-style-type: none"> • Applying algorithms such as <ul style="list-style-type: none"> ▪ swapping values ▪ finding aggregates ▪ isolate digits in an integer number ▪ finding the smallest/biggest of two numbers ▪ determine if a number is a factor of another number ▪ determine if a number is even • Comparing algorithms to built-in methods for performing same/similar tasks <p>String manipulation</p> <ul style="list-style-type: none"> • Basic string operators: concatenation and comparison • String manipulation using string methods: <ul style="list-style-type: none"> ▪ inserting and deleting characters ▪ determine the position of a character ▪ find a character/substring ▪ determine the length of a string • Extend applications by adding auxiliary methods to perform simple string manipulation in the form class <p>Solution development</p> <ul style="list-style-type: none"> • Design and develop solutions for specific problems that include computational thinking and applying software engineering principles using event-driven programming within the OOP paradigm <ul style="list-style-type: none"> ▪ Problems focusing on problem solving, OOP constructs and principles as well as algorithmic thinking 	<p>Explore algorithms for general computing problems, e.g.</p> <ul style="list-style-type: none"> • Determine whether a number is a even, odd number • Determine whether a number is a factor of another number • Lowest common multiple (LCM), greatest common divisor (GCD) • Determine current age given birth date <p>Explore algorithms for general string manipulation, e.g.</p> <ul style="list-style-type: none"> • Use ID number to determine age, gender
<p>Assessment (PoA):</p>	<p>Reporting</p>
<p>1 practical test + 1 theory test</p>	<p>Add raw marks and totals of the two tests and convert to percentage to determine term mark</p>

Grade 11: Term 2 – 10 weeks/40 hours, including examinations (2 weeks)	
Communication Technologies: Electronic Communications (±1 week/4 hours)	
<ul style="list-style-type: none"> • Mobile/wireless e-communication <ul style="list-style-type: none"> ▪ E-mail and blogging ▪ Micro blog, SMS, instant messaging ▪ Media: video casting, podcasting, VoIP, video conferencing • Mobile technology <ul style="list-style-type: none"> ▪ Mobile devices such as cell phones, smart phones, feature phones ▪ Mobile browser – description and examples • Wireless technologies <ul style="list-style-type: none"> ▪ Access points ▪ GPS, 3G, 4G, WiMAX, Bluetooth, etc. ▪ Difference in range and bandwidth (non-technical) • Protocols <ul style="list-style-type: none"> ▪ How protocols control data, e.g. POP3, SMTP, VoIP • Security <ul style="list-style-type: none"> ▪ Passwords, firewalls, encryption 	
Systems Technologies: Computer Management (±½ week/2 hours)	
Extend computer management issues regarding safeguarding against threats	
<ul style="list-style-type: none"> • Safety and security <ul style="list-style-type: none"> ▪ Human error (GIGO, accidents) • Threats <ul style="list-style-type: none"> ▪ Physical access <ul style="list-style-type: none"> ○ Theft ○ Flash drives and portable media ▪ Hardware failure <ul style="list-style-type: none"> ○ Storage ○ Power ▪ Network vulnerability <ul style="list-style-type: none"> ○ Virus, worm, Trojan, rootkit, spoofing, phishing • Remedies <ul style="list-style-type: none"> ▪ Backup, UPS, passwords, rights, firewalls, anti-virus, validation 	
Social Implications (±½ week/2 hours)	
<ul style="list-style-type: none"> • Social issues applicable to term 2 content such as social engineering, impact of social websites • List and discuss computer and human error and the effects thereof such as: <ul style="list-style-type: none"> ▪ Accuracy and validity – data input ▪ Data types used ▪ Verification and validation of data ▪ Software bugs ▪ Hardware failure ▪ Hardware configurations • Preventative actions 	
Solution Development: Software Engineering Principles and Problem solving (±1 week/4 hours)	Notes
<ul style="list-style-type: none"> • What is software development? • Planning and implementing a solution <ul style="list-style-type: none"> ▪ Define/understand the problem/task <ul style="list-style-type: none"> ○ Read the specifications and analyse the problem/task to determine the requirements ▪ Design the interface and the solution <ul style="list-style-type: none"> ○ Develop a logical solution based on the specifications and analysis as well as sound software engineering principles ○ Consider functionality and usability issues in designing the interface ▪ Code/implement <ul style="list-style-type: none"> ○ Incorporate suitable programming constructs in the development of a solution ▪ Test and debug the program <ul style="list-style-type: none"> ○ Use testing and debugging techniques and methods ▪ Document, implement and maintain the program • Planning techniques using any appropriate tools 	<p>Diagrams/visual tools for design purposes: Use any appropriate tools/techniques:</p> <ul style="list-style-type: none"> • TOE (Task, Objects, Events) charts • Noun-Verb analysis • IPO diagrams • UML: Use case diagram

Grade 11: Term 2 – 10 weeks/40 hours, including examinations (2 weeks)	
Solution Development: Application Development using a high-level programming language (±5 weeks/20 hours)	Notes
<p>Using good principles of algorithmic development and problem solving, extend programming with high-level language:</p> <p>Extend decision making</p> <ul style="list-style-type: none"> • Case/Switch construct <p>Simple Iteration</p> <ul style="list-style-type: none"> • Use simple iteration structures (<i>for</i>) <p>Arrays</p> <ul style="list-style-type: none"> • Arrays as a data structure (1-dim) <ul style="list-style-type: none"> ▪ Structure ▪ Step through items ▪ Basic operations, e.g. aggregates ▪ Searching using the linear search algorithm <p>Basic input validation techniques</p> <ul style="list-style-type: none"> • Input validation using code constructs <p>Input and output using a text file</p> <ul style="list-style-type: none"> • Apply simple file input and output using a text file to populate data structures and to develop simple reports <ul style="list-style-type: none"> ▪ Text files are incorporated utilising text stream operations and methods which load and save a file stream, etc. ▪ Accessing text stream operations and methods to load and save a file stream, etc. ▪ Utilise exceptions to catch errors on input and output <p>Conditional iteration</p> <ul style="list-style-type: none"> • Extend iteration <ul style="list-style-type: none"> ▪ Sentinel-controlled loops • Simple nested loops <p>Text-based reports</p> <ul style="list-style-type: none"> • Generating a simple text-based report, e.g. summary of data <p>Solution development</p> <ul style="list-style-type: none"> • Design and develop solutions for specific problems that include computational thinking and applying software engineering principles using event-driven programming within the OOP paradigm <ul style="list-style-type: none"> ▪ Problems focusing on problem solving, OOP constructs and principles and algorithmic thinking ▪ Incorporate software development principles 	<p>Lists: Built-in list objects could also be used.</p> <p>Studying generic algorithms develop logic and understanding and provide a foundation to apply similar principles and logic in developing solutions for specific problems.</p>
Assessment (PoA):	Reporting
1 test + 1 examination (1 practical paper + 1 theory paper)	Add raw marks and totals of the test and two papers and convert to percentage to determine term mark

Grade 11: Term 3 – 10 weeks/40 hours
Data and Information Management: Database Management ($\pm\frac{1}{2}$ week/2 hours)
<ul style="list-style-type: none"> • Describe database management software (DBMS) • Examples of DBMS software, e.g. <ul style="list-style-type: none"> ▪ Microsoft SQL Server ▪ Oracle ▪ Microsoft Access ▪ Blackfish ▪ Open source databases, e.g. PostgreSQL, MySQL • Database management – size and accessibility <ul style="list-style-type: none"> ▪ Desktop vs server (size and accessibility) ▪ Distributed (e.g. Google) ▪ DBMS software • Overview of database-related careers and roles of people involved <ul style="list-style-type: none"> ▪ DBA (database administrator) ▪ Unix administrators ▪ Programmers ▪ Analysts ▪ Project managers
Data and Information Management: Database design concepts (± 2 weeks/8 hours)
<ul style="list-style-type: none"> • Relationship between data, information, knowledge and decision making • Characteristics of quality data: <ul style="list-style-type: none"> ▪ Accuracy, correctness, currency, completeness, relevance ▪ Data validation, e.g. format check, data type check, range check, check digit • Qualities of valuable information • How to get to information <ul style="list-style-type: none"> ▪ Accessing and manipulating data <ul style="list-style-type: none"> ○ Manual ○ Electronic • Grouping data <ul style="list-style-type: none"> ▪ Records and fields ▪ Different types of fields and their purpose, e.g. primary key, alternate key ▪ Tables ▪ Relationships • Create a simple database with focus on table design without relationships • Data maintenance tasks such as <ul style="list-style-type: none"> ▪ Insert/add, delete, edit ▪ Process, sort, query (generating information from a database)
Social Implications ($\pm\frac{1}{2}$ week/2 hours)
<ul style="list-style-type: none"> • Social issues applicable to term 3 content • IT-related careers and the effects of digitalisation <ul style="list-style-type: none"> ▪ Careers: PC technician, programmer, network administrator, graphics design, web authoring, security consultants, systems analyst ▪ Effect on workplace and employment practices ▪ Mobile offices, virtual office, decentralisation of labour, office automation, robotics, artificial intelligence ▪ Ability to balance the advantages and disadvantages of a computerised system
Solution Development: Application Development using a high-level programming language ($\pm 4\frac{1}{2}$ weeks/18 hours)
<p>Using good principles of algorithmic development and problem solving, extend programming to incorporate database programming:</p> <ul style="list-style-type: none"> • Accessing a database through programming language constructs • Set up a connection or connect to a database (single table) by providing path in code statements • Develop a multi-form/multi-screen GUI incorporating simple controls – consider functionality and usability • Query a database (single table) using simple SQL constructs <ul style="list-style-type: none"> ▪ SELECT * (fields) FROM table_name WHERE criteria • Use programming language constructs in the execution of various simple database transactions <ul style="list-style-type: none"> ▪ Access fields and records within a dataset with code constructs and applicable methods ▪ Navigate the records of a dataset ▪ Modify individual fields and records within a dataset with code constructs and applicable methods, and apply all changes ▪ Manipulate a dataset object and records with code constructs and apply all changes

Grade 11: Term 3 – 10 weeks/40 hours	
<ul style="list-style-type: none"> ▪ Incorporate dataset event handlers and methods as part of the solution ▪ Reinforce concepts such as iteration and conditions • Use common dataset event handlers in the development of a solution • Reinforce methods as part of a solution • Apply simple parameter passing and return values using class methods as part of the form class • Design and develop solutions for specific problems that include computational thinking and applying software engineering principles <ul style="list-style-type: none"> ▪ Apply generic algorithms as part of the solution ▪ Incorporating database transactions managed by methods or events ▪ Devise a specific algorithm where applicable to solve a problem utilising user-defined code constructs or built-in methods ▪ Motivate the use of a specific algorithm ▪ Validate the solution against a set of data using different techniques, e.g. trace tables, watches, manual output comparison <p>Design and develop solutions for specific problems that include computational thinking and applying software engineering principles using event-driven programming within the OOP paradigm which may include database connectivity as part of the solution</p>	
Solution Development: Software Engineering Principles and PAT ($\pm 2\frac{1}{2}$ weeks/10 hours)	
<ul style="list-style-type: none"> • Start with Practical Assessment Task <ul style="list-style-type: none"> ▪ Reinforce problem-solving steps ▪ Reinforce software engineering principles 	
Assessment (PoA):	Reporting
1 practical test + 1 theory test	Add raw marks and totals of the two tests and convert to percentage to determine term mark

Grade 11: Term 4 – 10 weeks/40 hours, including examinations (2 weeks)
Internet Technologies: Internet and WWW (± ½ week/2 hours)
<ul style="list-style-type: none"> • Overview of the evolution of the Internet in terms of: <ul style="list-style-type: none"> ▪ Software and applications <ul style="list-style-type: none"> ○ WEB 1.0, WEB 2.0, WEB 3.0 ▪ Technology <ul style="list-style-type: none"> ○ Fixed location vs mobile • Overview of multimedia as part of Internet technologies <ul style="list-style-type: none"> ▪ Download vs streaming ▪ Live broadcasts ▪ Video on-demand and IPTV (Internet Protocol Television) • Media <ul style="list-style-type: none"> ▪ Compression technology (MP3, Mpeg4, Mpeg2, Jpeg) ▪ Compression: Quality vs bandwidth and speed
Internet Technologies: Internet Services Technologies (±1 week/4 hours)
<ul style="list-style-type: none"> • Overview of Internet services technologies <ul style="list-style-type: none"> ▪ Types of websites (i.e. what they offer) <ul style="list-style-type: none"> ○ Static vs dynamic sites (ability to store data, interactivity, media, advantages and disadvantages) ○ Location based services sites ○ Internet sites' accessibility to mobile devices • Overview of supporting technologies: <ul style="list-style-type: none"> ▪ HTTP, HTTPS, FTP, VoIP, RSS, SEO (search engine optimisation) ▪ Rich Internet applications ▪ Security services • Internet vs Intranet vs Extranet • Internet related careers <ul style="list-style-type: none"> ▪ Web designer ▪ Web author ▪ Graphics and multimedia designer
Data and Information Management: Database design concepts (±2 weeks/8 hours)
<ul style="list-style-type: none"> • Characteristics of a good database <ul style="list-style-type: none"> ▪ Data integrity ▪ Data independence ▪ Data redundancy ▪ Data security ▪ Data maintenance (ease of) • Problems with databases <ul style="list-style-type: none"> ▪ Anomalies ▪ How to get rid of anomalies (concept of normalisation) ▪ Split tables and create relations ▪ Key fields <ul style="list-style-type: none"> ○ Reinforce primary and alternate keys ○ Foreign keys ○ Composite keys ▪ Example of basic relationship enabled by the utilisation of key fields • Design guidelines • Design and create a relational database • Set up relationships between tables <ul style="list-style-type: none"> ▪ 1:M e.g. register class → pupils ▪ Two tables showing master detail relationship with at least one foreign key in one table ▪ Primary key and foreign key • Query a database using a join on a maximum of two tables with multiple criteria (the database may contain more than two tables, however a maximum of two tables is joined for query purposes) <ul style="list-style-type: none"> ▪ Querying a database (table) using a simple WHERE clause ▪ Querying a database using structured query language (SQL) – a single join utilising the WHERE clause • Simple entity relations diagrams (ERD)
Social Implications (±½ week/2 hours)
<ul style="list-style-type: none"> • Social issues applicable to term 4 content • Describe the influences of computer and mobile technologies on society due to globalising trends <ul style="list-style-type: none"> ▪ Online services (online banking, booking reservations, e-learning) ▪ Video conferencing, interactive whiteboards, online banking, cell phone banking, social websites (e.g

Grade 11: Term 4 – 10 weeks/40 hours, including examinations (2 weeks)	
Facebook)	
Solution Development: Application Development using a high-level programming language (±3 weeks/12 hours)	Notes
Extend array manipulation <ul style="list-style-type: none"> • Array (1-D) <ul style="list-style-type: none"> ▪ Sorting <ul style="list-style-type: none"> ○ Sorting an array using code constructs – any sorting algorithm ▪ Compare linear search and binary search algorithm Solution development <ul style="list-style-type: none"> • Use algorithmic thinking and software engineering principles to develop solutions for a variety of problems, focusing on computational problems which could include a database as part of the solution: <ul style="list-style-type: none"> ▪ Apply generic algorithms as part of the solution ▪ Devise a specific algorithm where applicable to solve a problem utilising user-defined code constructs or built-in methods ▪ Contrast generic algorithms to built-in methods ▪ Validate the solution against a set of data using different techniques, e.g. trace tables, watches, manual output comparison 	Problems should be solved using computational thinking and software engineering principles using event-driven programming within the OOP paradigm.
Solution Development: Software Engineering Principles and PAT (±3 weeks/12 hours)	
<ul style="list-style-type: none"> • Reinforce software engineering principles, algorithms and problem-solving techniques • Practical Assessment Task – finalise 	
Assessment (PoA):	Reporting
1 examination (1 practical paper + 1 theory paper) Plus Practical Assessment Task (PAT)	Convert: PAT to 25% Paper 1 to 25% Paper 2 to 25% Term 1 + Term 2 + Term 3 marks to 25%

Grade 12

Grade 12: Term 1 – 10 weeks/40 hours
Data and Information Management: Database design and concepts (±1 week/4 hours)
<ul style="list-style-type: none">• Explain and motivate relational database design<ul style="list-style-type: none">▪ Relational database overview<ul style="list-style-type: none">○ Normalisation (overview and purpose) to reduce data redundancy and limit data anomalies○ Where does un-normalised data come from?<ul style="list-style-type: none">♦ Analyse general documents, e.g. a till slip to identify possible data entities○ Design/entities, keys, record organisation• What is transaction processing system with regard to various database transactions?• Reinforce the characteristics of a good database in terms of the design<ul style="list-style-type: none">▪ Data integrity, data independence, data redundancy, data security, data maintenance
Data and Information Management: Database Management (±½ week/2 hours)
<ul style="list-style-type: none">• Caring for and managing data<ul style="list-style-type: none">▪ Value of data▪ How to protect data: validation, verification, integrity, logging changes (audit trail), warehousing, controlling access (passwords, security, user rights), parallel data sets• Hacking through data<ul style="list-style-type: none">▪ Invalid/false data▪ Database management software (DBMS) flaws (SQL injection)• Differentiate and list the roles of people as part of database management, and database systems development (roles, responsibilities)<ul style="list-style-type: none">▪ Database administrator (DBA)▪ Programmer
Systems Technologies: Hardware (±1 week/4 hours)
<ul style="list-style-type: none">• Mobile technologies<ul style="list-style-type: none">▪ Examples: Smart phones, laptops, tablets, netbooks, e-book readers▪ Advantages of mobility▪ Constraints<ul style="list-style-type: none">○ Battery life○ Size○ Computing power vs power consumption• Overview of factors influencing performance of a computer<ul style="list-style-type: none">▪ CPU (speed and multi processing)▪ Memory capacity (cache and RAM)▪ Storage speed▪ Network speed• Motivate a typical computer system in respect of the hardware needed for a specific purpose<ul style="list-style-type: none">▪ Computer system for<ul style="list-style-type: none">○ Home/personal use○ Game and entertainment○ SOHO (Small-Office-Home-Office) user○ Power user
Communication Technologies Networks (±½ week/2 hours)
<ul style="list-style-type: none">• Setting up a network<ul style="list-style-type: none">▪ Essential parts<ul style="list-style-type: none">○ Switch, cables, wireless base station▪ Connecting to the Internet<ul style="list-style-type: none">○ Router/modem, ADSL/Wimax/3G▪ All-in-one solution ('router' is modem, router, switch and base station – all in one)• Sharing concepts<ul style="list-style-type: none">▪ Sharing files and folders, user rights, BitTorrent (Risks and benefits)▪ Online services (e.g. drop box/Mobile Me)• Remote access<ul style="list-style-type: none">▪ On local network, through Internet
Communication Technologies: E-communications (±½ week/2 hours)
<ul style="list-style-type: none">• Overview of security concepts<ul style="list-style-type: none">▪ Encryption▪ SSL (private and public key)▪ Certificates and security

Grade 12: Term 1 – 10 weeks/40 hours	
Social Implications (±½ week/2 hours)	
<ul style="list-style-type: none"> • Social issues applicable to term 1 content, e.g. reducing the environmental impact of the use of computers could be reduced • Discuss various ways to stay informed about computer technology • Getting latest product upgrades, viruses and other threads, upgrading 	
Solution Development: Application Development using high-level programming language (±4 weeks/16 hours)	Notes
<p>Reinforce concepts, programming skills, algorithms and problem-solving skills developed in Grades 10 and 11 by means of application development</p> <ul style="list-style-type: none"> • Develop a simple user-defined class to meet the program specifications as part of the solution • Instantiate a user-defined object as part of the solution • Reinforce method invocation • Differentiate between various types of methods in relation with their use and purpose (constructors, destructors, accessors, mutators, auxiliary) <p>Extend database and programming to incorporate relational databases</p> <ul style="list-style-type: none"> • Reinforce concepts and techniques mastered in Grade 11 • Accessing a relational database through a programming language • Set up a connection or connect to a relational database by providing path in code statements • Query a database using a join on a maximum of two tables with multiple criteria (the database may contain more than two tables, however a maximum of two tables is joined for query purposes) – reinforce: <ul style="list-style-type: none"> ▪ Querying a database using structured query language (SQL) – a single join utilising the WHERE clause • Use programming language constructs in the execution of various simple database transactions • Develop a multi-form GUI incorporating simple controls • Share data amongst forms as part of the solution • Use appropriate algorithms and/or built in methods in the manipulation of data such as sorting routines, string based routines, date and time • Incorporate defensive programming techniques as part of the solution (data validation) <ul style="list-style-type: none"> ▪ Check for empty fields, ranges, valid formats, data • Design and develop code constructs to generate a text based report • Construct more complex algorithms, e.g. remove duplicates from a list/table • Develop solutions for various problems using computational thinking and applying software engineering principles that include both database and non-database problems <ul style="list-style-type: none"> ▪ Test and validate a solution against a set of design specifications ▪ Alter a solution to meet a set of design specifications ▪ Document a solution design and development ▪ Motivate the design and development of the solution ▪ Evaluate a solution against other solutions 	<p>The scope of OOP</p> <ul style="list-style-type: none"> ▪ Design a simple class with basic attributes and simple methods ▪ Small modifications to a given class such as completing a method, adding one or more methods or adding/ modifying current attributes to a class provided, call methods from an instance of a class <p>Algorithms Removing duplicates from a list/table/array</p>

Solution Development: Software Engineering Principles and PAT (±2 weeks/4 hours)	Notes
<ul style="list-style-type: none"> • Overview and comparison of different methodologies such as waterfall, rapid application development (RAD), incremental and agile • Start with PAT – Task description and analysis of requirements using an appropriate methodology 	Diagrams/visual tools for design purposes: Use any appropriate tools/techniques: <ul style="list-style-type: none"> • TOE (Task, Objects, Events) charts • Noun-Verb analysis • IPO diagrams • CRC cards • UML: Use case diagram, basic class diagram
Assessment (PoA):	Reporting
1 practical test + 1 theory test	Add raw marks and totals of the two tests and convert to percentage to determine term mark

Grade 12: Term 2 – 10 weeks/40 hours, including examinations (2 weeks)	
Systems Technology: Software (±1 week/4 hours)	
<ul style="list-style-type: none"> • Overview of cloud computing and virtualisation <ul style="list-style-type: none"> ▪ Describe cloud computing ▪ Effect on hardware needs ▪ Software as a service (SaaS) <ul style="list-style-type: none"> ○ Description and advantages ○ Who owns what? ▪ Virtualisation of servers • Arguments for and against 	
Systems Technologies: Computer Management (±½ week/2 hours)	
<ul style="list-style-type: none"> • Factors influencing computer management • Recommend management tasks for general housekeeping and to maintain data integrity and protect the system 	
Social Implications (±½ week/2 hours)	
<ul style="list-style-type: none"> • Computer criminals <ul style="list-style-type: none"> ▪ Hackers, crackers, cyber gangs, virus authors • Types of cyber crimes • Effect of cyber crimes • Computer crimes such as hardware, software, information, identity theft, bandwidth theft, theft of time and services <ul style="list-style-type: none"> ▪ Internet-related fraud scams ▪ Internet attacks (worms, virus, denial of service, back doors) ▪ Phishing ▪ Unauthorised remote control and administration, e.g. botnets, zombies ▪ Right to access vs right to privacy, misuse of personal information • Safeguards against computer crimes, threats and criminals 	
Solution Development: Application Development using a high-level programming language (±4 weeks/10 hours)	
<ul style="list-style-type: none"> • Arrays as a data structure (2-dim) <ul style="list-style-type: none"> ▪ Structure ▪ Step through items ▪ Basic operations, e.g. row/column aggregates <p>Extend database and programming</p> <ul style="list-style-type: none"> • Design and develop a solution incorporating SQL <ul style="list-style-type: none"> ▪ Select, distinct ▪ Insert, update, delete ▪ Where ▪ Order by ▪ Group by ▪ Special operators: Between, In, Like, Is Null, Having ▪ Creating calculated fields, concatenating fields ▪ Formatting with round, int, etc. ▪ Casting a field ▪ Create a join query (single joins) using ‘Where’ ▪ Mathematical operators ▪ Aggregate functions: Sum, Average, Min, Max, Count ▪ Common date functions ▪ String functions (Length, Mid, Left, Right) ▪ Simple sub queries • Use algorithmic thinking and software engineering principles to develop solutions for a variety of problems, that include both database and non-database problems 	
Solution Development: Software Engineering Principles and PAT (±2 weeks/8 hours)	
<ul style="list-style-type: none"> • Reinforce software engineering principles • Interface design: Functionality and usability principles and program design • Practical Assessment Task – continue 	
Assessment (PoA):	Reporting
1 test + examination (1 practical paper + 1 theory paper)	Add raw marks and totals of the test and two papers and convert to percentage to determine term mark

Grade 12: Term 3 – 10 weeks/40 hours, including examinations (3 weeks)
Data and Information Management: Database Design Concepts (±½ week/2 hours)
<ul style="list-style-type: none"> • Data collection – Overview and examples <ul style="list-style-type: none"> ▪ RFID ▪ Online ▪ Invisible (e.g. credit card, loyalty card, government, forms, toll road passes, cellphone) • Data warehousing <ul style="list-style-type: none"> ▪ Describe data warehousing ▪ Purpose and uses • Data mining – description and purpose <ul style="list-style-type: none"> ▪ SQL ▪ Extracting data ▪ Looking for patterns ▪ Discovering knowledge ▪ Strategy • Location-based data
Internet Technologies: Internet and WWW (±½ week/2 hours)
<ul style="list-style-type: none"> • Trends and emerging technologies, e.g. <ul style="list-style-type: none"> ▪ WEB 3.0 (Semantic Web) • Online applications and storage • Improve searching <ul style="list-style-type: none"> ▪ Semantic search ▪ Mediated search
Internet Technologies: Internet Services Technologies (±½ week/2 hours)
<ul style="list-style-type: none"> • Online applications <ul style="list-style-type: none"> ▪ Storing data <ul style="list-style-type: none"> ○ Locally (cookies) ○ Online (databases) ○ Role of SQL, scripting languages (e.g. PHP, JavaScript), XML ▪ Running instructions <ul style="list-style-type: none"> ○ Locally (scripts, AJAX) ○ Online (server side, scripts and code) ▪ Formatting output <ul style="list-style-type: none"> ○ CSS
Social Implications (±½ week/2 hours)
<ul style="list-style-type: none"> • Social issues applicable to term 3 content • Explain how computers provide solutions to issues of national and international importance such as: <ul style="list-style-type: none"> ▪ Distributed computing power ▪ Decision making • Describe the evolution of social networking and the effect on society: <ul style="list-style-type: none"> ▪ Information overload ▪ Availability of personal information <ul style="list-style-type: none"> ○ Consequences of search engines and group communications ○ Social, political, environmental ○ Global community – cultural effects ○ Social websites and social engineering ○ Wikis • List and discuss issues regarding privacy and information sharing <ul style="list-style-type: none"> ▪ Cookies, anonymity, Global Unique Identifiers, file sharing – movies, music
Solution Development: Application Development using a high-level programming language (±2 weeks/8 hours)
<ul style="list-style-type: none"> • Consolidate and reinforce content, concepts and skills • Design and develop solutions for a variety of problems that include computational thinking and applying software engineering principles <ul style="list-style-type: none"> ▪ Test and validate a solution against a set of design specifications ▪ Alter a solution to meet a set of design specifications ▪ Document a solution design and development ▪ Motivate the design and development of the solution ▪ Evaluate a solution against other solutions

Grade 12: Term 3 – 10 weeks/40 hours, including examinations (3 weeks)	
Solution Development: Software Engineering Principles and PAT (±3 weeks/12 hours)	
<ul style="list-style-type: none"> • Reinforce software engineering principles • Practical Assessment Task – Finalise 	
Assessment (PoA):	Reporting
1 test + examination (1 practical paper + 1 theory paper)	Add raw marks and totals of the test and two papers and convert to percentage to determine term mark

Grade 12: Term 4 – 10 weeks/40 hours, including examination (7 weeks/28 hours)	
Content using Case Studies – All topics (±1½ weeks/6 hours)	
Consolidate content, concepts and skills using case studies to:	
<ul style="list-style-type: none"> • Identify the basic hardware configuration of a computer in terms of the processor, memory and hard drive size • Understand computers and their uses • Know how to use computers as tools to access information and to communicate with others around the world • Make better buying decisions – interpret advertisements and make judgements about quality and usefulness when buying equipment and software • Know how to fix simple computer problems and deal with challenges that arise with utilising computers (and know when to call for help) • Know what kind of computer uses could benefit and advance work place and career path opportunities • Know how to protect themselves against online villains and threats • Know how to apply digital tools to communicate, gather, analyse, use information and solve problems • Understand technology concepts, systems and operations • Recommend specific hardware/software for a specific scenario 	
Solution Development: Application Development (±1½ weeks/6 hours)	
Consolidate content, concepts and skills to develop a software solution	
External examination (±7 weeks/24 hours)	
• Practical examination	25%
• Theory examination	25%
External examination: 1 practical paper + 1 theory paper plus Practical Assessment Task (PAT)	

Section 4

Assessment in Information Technology

4.1 Introduction

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 in order to improve the process of learning and teaching.

Assessment involves activities that are undertaken throughout the year. In grades 10 – 12 assessment comprises two different but related activities: informal daily assessment (assessment for learning) and formal assessment (assessment of learning).

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

4.2 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' progress. This is done through observation, discussion, practical demonstrations, learner-teacher conferences, informal classroom interactions, etc. Informal assessment may be as simple as stopping during the lesson to observe learners or to discuss with learners how learning is progressing. Informal assessment should be used to provide feedback to the learners and to inform planning for teaching, but need not be recorded. It should not be seen as separate from learning activities taking place in the classroom. Learners or teachers can mark these assessment 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.

4.3 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 Information Technology:

Grade 10 and 11

Formal Assessment			
During the Year	End-of-Year Examination		
25%	75%		
SBA tasks	Practical Assessment Task	End-of-Year Exam Papers (50%)	
25%	25%	25%	25%
<ul style="list-style-type: none"> • 5 tests • 1 exam (mid-year) 	Project Software development project including aspects of planning cycle as well as principles of software engineering	Written exam 2 – 3 hours Theory aspects of all content, concepts and skills of all topics	Practical exam 3 hours Solution Development

Grade 12

Formal Assessment			
During the Year	End-of-Year Examination		
25%	75%		
SBA	Practical Assessment Task	End-of-Year Exam Papers (50%)	
25%	25%	25%	25%
<ul style="list-style-type: none"> • 4 tests • 2 exams (mid-year and trial) 	Project Software development project including aspects of planning cycle as well as principles of software engineering	Written exam 3 hours Theory aspects of all content, concepts and skills of all topics	Practical exam 3 hours Solution Development

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.

4.3.1 Types of formal assessment for Information Technology

Project

A project assesses the learner's ability to apply knowledge, skills and a range of competencies in an integrated manner, many of which cannot be assessed in other ways. It has a degree of open-endedness, but is focused and results in individual but similar tasks. The time to complete a project normally ranges from a few days to several weeks.

In IT the project is the practical assessment task (PAT).

The project should enable a learner to apply a combination of techniques, knowledge and skills to new situations to complete the task or accomplish a goal. It should also encourage learners to use and combine information, data, and ideas to solve problems, discover and explain patterns, relationships or trends and predict behaviour/events.

A project should require the learner to:

- do some planning/preparation/investigation/research/data gathering to solve the identified problem/task;
- perform the task/carry out instructions (according to criteria given);
- produce a product such as a software application (this could include a limited number of smaller products such as a planning document, that builds up to the final product, which the teacher could monitor or assess informally or formally);
- demonstrate thinking and decision making skills; and
- demonstrate some innovation and creativity.

To set and manage the project, the teacher should:

- determine the content/skills/knowledge to be addressed;
- set clear criteria and give clear instructions to guide the learner (the learner should know exactly what to do and what is expected);
- keep the scope manageable;
- determine which resources will be required to complete the project and ensure that learners have access to these resources;
- determine the time frame/duration/due date;
- determine mark distribution and compile an assessment tool; and
- continuously monitor the completion of the project and guide the learners.

Tests

A test could be a practical test or a written test. The programme of assessment should reflect a balance between practical and written tests. Tests could include open book tests.

A test for formal assessment should not comprise of a series of small tests, but should cover a substantial amount of content and the duration should be 45 – 60 minutes.

Open book tests require learners to find information and apply knowledge and skills. Learners are tested on understanding and application of learning material and not on rewriting. Open book tests should not include only short questions. They must include questions/tasks that will encourage thinking and decision making.

For written open book tests, learners are required to write longer reflective answers, such as paragraph type responses to a given scenario, e.g. case studies. Paragraphs providing reasons and supporting evidence/arguments are essential.

For practical open book tests learners are required to apply a combination of a series of procedures and techniques to new situations in order to provide a specific answer or accomplish a specific goal, e.g. integrated practical tasks that encourage computational thinking.

Each test, open book test and examination must reflect different cognitive levels.

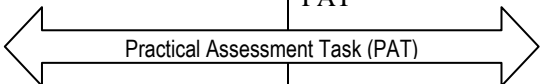
Formal assessments must cater for a range of cognitive levels and abilities of learners as shown in the table below:

Lower order (Knowledge/remembering) (Routine procedures)	Middle order (Understanding/applying) (Multi-step procedures/Extension)	Higher order (Analysing/evaluating/creating) (Problem solving)
30%	40%	30%

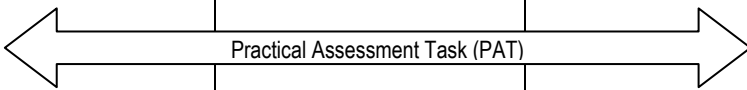
4.4 Programme of Assessment

The following tables provide the programme of assessment requirements for Information Technology:

Grade 10 and 11

Programme of Assessment			
SBA per Term			
Term 1: 1 Practical Test + 1 Theory Test	Term 2: 1 Test + 1 Examination comprising 2 Papers: 1 Theory + 1 Practical	Term 3: 1 Practical Test + 1 Theory Test	Term 4: 1 Examination comprising 2 Papers: 1 Theory + 1 Practical Plus PAT
			
Term Mark (Terms 1 – 3): <ul style="list-style-type: none"> Each term, add raw marks and totals and convert to % for term mark 			
Promotion Mark: <ul style="list-style-type: none"> Add raw marks and totals for assessment tasks from term 1 to term 3 and convert to 25% Convert PAT mark to 25% Convert Paper 1 to 25% Convert Paper 2 to 25% 			

Grade 12

Programme of Assessment			External Assessment
SBA per Term			
Term 1: 1 Practical Test + 1 Theory Test	Term 2: 1 Test + 1 Examination comprising 2 Papers: 1 Theory + 1 Practical	Term 3: 1 Test + 1 Examination comprising 2 Papers: 1 Theory + 1 Practical	Term 4: 1 External Examination comprising 2 Papers: 1 Theory + 1 Practical Plus Practical Assessment Task
			
Term Mark (Terms 1 – 3): <ul style="list-style-type: none"> Each term, add raw marks and totals and convert to % for term mark SBA Mark: <ul style="list-style-type: none"> Add raw marks and totals for assessment tasks from term 1 to term 3 and convert to 25% External Examination: <ul style="list-style-type: none"> Convert Paper 1 to 25% Convert Paper 2 to 25% Convert PAT to 25% 			

4.4.2 Examinations

Practical Assessment Task (25% of the total marks for the subject)

The IT PAT assesses the learners' ability to develop a solution for a specific task using the software development tools studied in Grades 10 – 12.

Learners should apply appropriate problem-solving techniques and software engineering principles in developing the application.

The IT PAT comprises different components/stages that represent the software development process using any appropriate approach/methodology. Software development activities typically include aspects such as:

- planning (understanding the problem/task and identifying the requirements);
- design (interface and program design using appropriate design tools and techniques – learners will not be expected to use any specific software design tool); and
- coding, testing, implementation and internal documentation.

In Information Technology the PAT counts 25% of the total promotion/certification mark for the subject. It is implemented throughout the school year and should be undertaken as one extended task, which is broken down into different phases or a series of smaller activities.

Each task must include a declaration of authenticity.

In Grade 12, the criteria for the Practical Assessment Task are externally set, internally administered and marked and externally moderated.

The topic of the PAT will be provided to schools each year by the end of the previous year.

Paper 1: 3-hour practical paper of 150 marks (25% of the total marks for the subject)

This will be a practically oriented paper covering questions on Solution Development.

To successfully complete this paper, each learner must have access to his or her own computer in the exam room. Provision needs to be made for sufficient computers to enable the examination to be completed in **2 sittings**.

This paper assesses the practical skills as well as the knowledge and understanding underlying the skills pertaining to Solution Development, i.e. the high-level programming language studied (i.e. excluding the introductory graphical programming tool used in Grade 10) which includes interaction with a database.

The questions will be based around a scenario.

The paper will comprise questions covering the following broad topics:

- OOP-based problem solving
- Integrated data-aware solution that will also include problem-solving as part of the solution
- General problem solving

Software design techniques, methods and tools such as UML, CRC cards, etc. will not be examined as part of the practical paper.

The learner will not be required to enter large amounts of data. The required data could be retrieved from the data disk or imported from documents such as a text file, or a database table. All GUIs will be provided.

Marks for questions must be allocated towards concepts, constructs (groupings of code concepts) and problem solving techniques, e.g. application of an iteration structure as part of the solution (correct structure) as well as for the correct use of the structure.

Paper 2: 3-hour written paper of 150 marks (25% of the total marks for the subject)

The paper will cover all theory aspects of all content, concepts and skills of topics, including elements of Solution Development, e.g. algorithmic development, data structures, program design and general programming concepts as well as generic problem solving questions.

The questions will be based around a scenario.

The following format could be used:

Section		Description
A	Human Computer Interaction and Social Implications These topics could be integrated as part of the other sections and will not be a separate section in the paper.	Short questions (±25 marks) A range of short questions covering all topics that could include <ul style="list-style-type: none"> • multiple-choice and • modified true and false.
B		Systems Technologies (±20 marks) Questions related to the content, concepts and skills in the Systems Technologies topic.
C		Communications Technologies and Network Technologies (±25 marks) Questions related to the content, concepts and skills in the Communication Technologies and Network Technologies topic.
D		Data and Information Management (±20 marks) Questions related to the management of data and the concept of information management.
E		Solution Development (±20 marks) Questions aligned to the Solution Development topic which assess the knowledge and understanding underlying the concepts and skills in the Solution Development topic.
F		Integrated Scenario (±40 marks) This section is based on a single large-scale scenario and assess all the topics.

Software design tools for examination purposes as part of the theory paper are limited to basic flow charts, class diagrams and use case diagrams.

Content to be covered

Assessment addresses the content as set out in this document. Due to the conceptual progression of the content across the grades, content and skills from Grades 10 – 12 will be assessed in the external papers at the end of Grade 12.

A list of emerging technologies to be covered for examination purposes will be provided to schools each year by the end of the previous year.

4.5 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 or his readiness to progress or be 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 a number of ways. These include report cards, parents' meetings, school visitations, parent-teacher conferences, phone calls, letters, class or school newsletters, etc. Teachers in all grades report in percentages against the subject.

7 levels of competence have been described for each subject listed for Grades R – 12. The various achievement levels and their corresponding percentage bands are as shown in the table below:

Codes and percentages for recording and reporting

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

Teachers will record actual marks against the task by using a record sheet; and report percentages against the subject on the learners' report cards.

4.6 Moderation of assessment

Moderation refers to the process that ensures that the assessment tasks are fair, valid and reliable. Comprehensive and appropriate moderation practices must be in place for the quality assurance of all subject assessments.

4.6.1 Formal assessment (SBA)

- Grade 10 and 11 tests and examinations are internally moderated. The subject advisor must moderate a sample of these tasks during his/her school visits to verify the standard of tasks and the internal moderation.
- Grade 12 tests and examinations must be moderated at provincial level. This process will be managed by the provincial education department.
- Subject advisors must moderate samples of tests and examination papers before they are written by learners to verify standards and guide teachers on the setting of these tasks.

4.6.2 Practical Assessment Task (PAT)

- Grade 10 and 11: Teachers assess the practical assessment tasks in Grades 10 and 11. The subject advisor must moderate a sample of PATs during his/her school visits to verify the standard of tasks and the internal moderation

- Grade 12: Teachers assess the practical assessment tasks according to the externally set assessment tool. The subject advisor must moderate a sample of each phase of the PATs during his/her school visits to verify the interpretation of the assessment tool and the standard of marking. Completed PATs must also be moderated at provincial level. This process will be managed by the provincial education department.

4.7 Annexures

Annexure A – Glossary of acronyms and abbreviations

4.8 General

This document should be read in conjunction with:

4.8.1 *National policy pertaining to the programme and promotion requirements of the National Curriculum Statement Grades R – 12; and*

4.8.2 The policy document, *National Protocol for Assessment Grades R – 12.*

Annexure A

Glossary of Acronyms and Abbreviations

1:M	One-to-many
1-D	One-dimensional
3G	Third generation of cellular wireless
4G	Fourth generation of cellular wireless
ADSL	Asymmetric Digital Subscriber Line
BIOS	Basic Input Output System
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DBA	Database Administrator
DBMS	Database Management System
EDP	Event Driven Programming
ERD	Entity Relationship Diagrams
FOSS	Free Open Source Software
FTP	File Transfer Protocol
GIGO	Garbage-In Garbage-Out
GPS	Global Positioning System
GUI	Graphical User Interface
HAN	Home Area Network
HCI	Human Computer Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
I/O	Input-Output
ICT	Information and Communication Technology
IDE	Integrated Development Environment
IP	Internet Protocol
IPO	Input-Processing-Output
IPTV	Internet Protocol Television
IT	Information Technology
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
MP3	MPEG-1 Audio Layer-3
MPEG	Motion Picture Expert Group
NIC	Network Interface Card
OOP	Object-Oriented Programming
OS	Operating System
PAN	Personal Area Network
PAT	Practical Assessment Task

PC	Personal Computer
PHP	Hypertext Preprocessor
PnP	Plug-and-Play
PoA	Programme of Assessment
POP3	Post Office Protocol
PoS	Point-of-Sale
RAD	Rapid Application Development
RAM	Random Access Memory
RFID	Radio-Frequency Identification
ROM	Read-Only Memory
RSS	Really Simple Syndication
SaaS	Software as a Service
SEO	Search Engine Optimisation
SMTP	Simple Mail Transfer Protocol
SMS	Short Message System
SOHO	Small Office Home Office
SQL	Structured Query Language
SSL	Secure Socket Layer
TOE	Task-Objects-Events
UML	Unified Modelling Language
URL	Uniform Resource Locator
USB	Universal Serial Bus
VoIP	Voice over Internet Protocol
VPN	Virtual Private Network
W3C	World Wide Web Consortium
WAN	Wide Area Network
Wi-Fi	Wireless Fidelity
WiMAX	Worldwide Interoperability for Microwave Access
WWW	World Wide Web
WYSIWIG	What You See Is What You Get
XML	Extensible Markup Language