Curriculum for Computing
(Computer Studies)

For

Track 2 - Preparation of Students for Paper 2B
&
Track 3 – Preparation of Students for Paper 2A

Revised June 2011
For SEC examinations from 2013

The order in which topics are here presented is not necessarily the order they should be covered in.

Teachers are encouraged to visit the official SEC Computing 2013 syllabus for:

- ***the level of detail required (available in the supplementary notes);***
- ***other relevant details which they may need for the proper and efficient conduct of the course.***

Contents:

Foreword

This revised Computing (Computer Studies) curriculum reflects the major changes that have been published in the SEC 2013 syllabus. These include the new subject title, the introduction of the Java programming language and the requisite of one coursework exercise. The curriculum caters for both Track 2 and Track 3 streams. Track 2 stream will eventually lead to SEC Paper 2B and the content to be covered **excludes** the grayed out areas of this curriculum. On the other hand Track 3 stream leads to SEC Paper 2A and the tutor should cover this curriculum in its entirety, i.e. **inclusive** of all grayed out sections.

During the revision we thought we should make the curriculum better reflect the SEC syllabus by replicating the text in the SEC syllabus, verbatim. However at times it was felt that there were some 'missing links' in the syllabus. With this in mind, the items in italicized text were appended to consolidate and/or clarify the expectations of the topic. It is suggested that certain topics within a Form be covered with other related items in that particular Form, so that there shall be a more holistic approach to the topic. A typical example of this is complementing the theoretical topics being covered with hands-on exercises during the practical sessions. On the same note, when it comes to problem solving and programming (and provided that you have access to the lab for the four scheduled lessons), most teachers have found it more fruitful and effective to dedicate all the four lessons on this topic.

This curriculum introduces the Lego Mindstorms kits to enhance the problem solving skills and programming concepts of our students. It is being iterated that the kits do not form part of the SEC syllabus and they should only be used to motivate students when introducing the above-mentioned skills. In this respect little emphasis is to be given to the perspective of robotics, therefore complicated and complex contraptions should be avoided. It is suggested that with Form 3s the use of the kits be scheduled for the second semester.

To keep with our traditional weightings for the annual examinations, i.e. 85% for the written paper and 15% for the practical assessment, a new setup was developed for the practical assessment mark.

- Form 3 students shall be assigned an exercise using the Lego Mindstorms kits and the marks awarded shall be their practical assessment component for the annual examination.
- The Form 4 practical component for the annual examination may be the foundation for the SEC coursework.
- The practical assessment mark for Form 5 students shall be the official coursework mark that will be forwarded to the MATSEC board.

The introduction of this new setup for Forms 3 and 4 brought with it the important issue of standardization of the marking process of the practical component across all schools. With this in mind this curriculum contains details of the marking criteria for both Form 3 and Form 4. The marking sheet and the requested documentation of each student must be readily available in **hardcopy format** for moderation

purposes. Moderation is an exercise to help and support the teacher, and, the Heads of Department and myself shall conduct this exercise.

Teachers teaching Form 5 students are reminded to allocate enough time for SEC past examination papers. I am sure that all teachers agree with me that the performance of most students greatly ameliorates during this experience.

I take this opportunity to wholeheartedly thank the Heads of Department - Marlene Galea, Joseph Vella and Robert Vella for their contribution, help and support in compiling this curriculum.

Godwin Zammit
Education Officer – Computer Studies

Background

The Computing curriculum was developed for students in Forms 3 to 5 as one of the elective subjects in the state secondary sector. The subject was introduced in scholastic year 2001-2 with the basic assumption that these students have no prior knowledge of computers. The curriculum caters for all students irrespective of gender, belief, ethnic, social and cultural background.

Due to the dynamic and changing nature of computer technology, the curriculum needs to be reviewed and evaluated periodically in order to reflect and suit (1) the technological influences and trends, and (2) the relative Matsec syllabus.

Though it is assumed that students taking the subject have no prior knowledge of computers, the diverse level of computer skills of students that are present in schools must be taken in consideration by teachers.

The National Minimum Curriculum

The National Minimum Curriculum recognises right from the start that developments in science and technology are among the greatest and most important challenges that it needs to address. The Curriculum emphasises that schools should help children and youngsters understand the impact that technology is having on all aspects of our lives and on the organisation of work. It also repeatedly stresses the need that students acquire proficiency in all aspects of literacy and numeracy and that they can make good and confident use of technology.[1]

Active and Effective Teaching and Learning

To promote effective learning, teachers are encouraged and expected to incorporate the following teaching strategies into their repertoire.

- Students should be encouraged to discover knowledge for themselves and classroom activities should be structured so that students can construct their own learning. These constructivist practices are to encourage students to be active learners and to be in control of their own learning.
- Collaborative learning should be encouraged, where students can share ideas during discussions. Through collaboration students can help each other with their learning, develop esteem and confidence and foster social interactions.
- Class activities should be structured to promote independent learning by students. Furthermore, they should promote critical thinking and reflection by students on what they have learnt.

Aims of the syllabus

The teaching of Computer Studies (now called Computing) aims at helping the students become competent, independent, technologically-oriented individuals and valid

[1] From the 'Foreword' by Ms Mary Vella Director Curriculum Management (ret) when the subject was introduced in 2001-2.

members of society. The students can then achieve lifelong benefits from their education. In order to reach this target, the syllabus intends to[2]:

- Stimulate and foster an interest in the use of computers;
- Develop practical skills in the use of computers;
- Develop skills in creatively applying information processing technology to problem solving;
- Provide a broad view of the range and variety of computer systems and applications;
- Develop the ability to communicate and interpret information and concepts relevant to computing;
- Introduce students to the fundamental concepts of computer science;
- Serve as a basis for further studies in Intermediate and Advanced Levels in Computing or Information Technology.

Objectives of the syllabus

The learner shall appreciate the importance of Computing as an instrument of integrating oneself into today's digital society by gradually developing the necessary ability and skills related to the use and workings of computers but in particular the use of technology in problem solving. The Computing programme has the objective of helping the learner to acquire the abilities and skills to[3]:

- Holistically enhance the learner's intellectual and social development;
- Be aware of the widespread use of computers in processing information;
- Use and/or recognise common computer applications;
- Be familiar with the main hardware functional units of a computer system and how these fit together and communicate to form a complete system;
- Understand how data is represented digitally;
- Differentiate between various types of complex computer systems such as those in airline reservation, banking, etc;
- Understand, analyse, design, code and test solutions to simple problems;
- Be knowledgeable of the major areas of computer applications;
- Develop awareness of the vast amount and variety of computer software available for different applications;
- Have an understanding of the practical problems in using computers;
- Enhance his/her educational and vocational prospects while understanding the potential of the subject for his/her future.

Assessment

Assessment is the process of gathering meaningful information which is used to make judgements on aspects of the learning cycle such as learners' performance against the achievement objectives, and the quality and effectiveness of learning programmes. It is important that:

[2] SEC Computing 2013 syllabus with kind permission of Matsec board.
[3] SEC Computing 2013 syllabus with kind permission of Matsec board

- A range of assessment procedures is used to provide useful information on students' progress against the achievement objectives stated in the curriculum;
- Assessment and evaluation are ongoing and help improve the ways Computing programmes are meeting the students' needs;
- Students are involved in the assessment of their own progress when learning the subject.

Effective assessment needs to:

- Be valid, reliable and authentic;
- Help students and teachers focus better on learning;
- Be rewarding in terms of offering guidance about progress;
- Give feedback on how and why a student understands or misunderstands and what direction the student must take to improve.

Formative and Summative assessment

Formative assessment should be given its due importance throughout the scholastic year. With formative assessment, students receive information about their performance so that they can learn more effectively. Formative assessment implies giving priority to understanding over memorisation and, to quality of work over quantity. Formative assessment offers the student a personalised evaluation of his/her performance.

During each scholastic year, two main summative evaluations are held, one in February (half yearly exam) and the other in June (annual exam). Where it is deemed necessary, after the February scripts are marked, students should be given feedback about their performance, preferably in the form of a correction of the exam paper. Unfortunately this cannot be done for the annual exam session.

Form 3

Learning Outcomes

By the end of Form 3 the learner shall be able to:

- Use the appropriate computing terminology in the correct context;
- Demonstrate the tasks performed by a computer system;
- Sketch and identify the components of a computer system
- Differentiate between analogue and digital systems and provide examples;
- Apply his/her knowledge of decimal, binary and hexadecimal number systems in number conversion;
- Identify the units of data storage and the representation of text;
- Apply the principles of the 3 basic logic gates to simple logic circuits;
- Identify and differentiate between electronic, magnetic and optical storage media/devices and provide relevant examples;
- Differentiate between serial and direct access, provide examples of typical media and demonstrate their application
- Demonstrate the differences between the various input devices and provide examples of areas of application;
- Demonstrate the differences between the various output devices and provide examples of areas of application;
- Apply basic principles of problem solving – definition and analysis of problem and design of solution using appropriate tools;
- Implement solutions to simple problems using the Mindstorms kits;
- Identify and use the basic features of an operating system;
- Use utility software, web browser and email software;
- Use a word processor, spreadsheet and a graphics package in specific situations.

FORM 3 – THEORETICAL COMPONENT			
Part	**Topic**	**Ref**	**Detail**
1	**COMPUTER APPLICATIONS**		
	The Computer System	1.1	The computer system as an information processing machine. Its tasks of handling information: input, process, output, store, retrieve, send and receive. *Data and Information.* *Hardware and Software. Peripherals.*
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	Components of a Computer System	2.2.1	CPU, I/O Sub-system, main and backing store. Flow of data and control between the main units using buses. *The block diagram should include I/O devices. At this instance in time the I/O sub-system should be treated as a 'black box' and flow lines indicate the flow of data.*
	Computer Logic	2.2.3	Distinction between analogue and discrete (*digital)* processes and quantities. Conversion of analogue quantities to digital form using sampling techniques. Use of 2-state electronic systems (logic 0 and logic 1) for reliability. *Examples of digital and analogue systems/processes.*
	Number Systems	2.1.1	Representation of numbers in binary. Conversion between decimal and binary, and between binary and hexadecimal. Use of subscripts 2, 10 and 16 for bases. *Also to include conversion between hex and decimal. The MSB and LSB*
	Coding Systems	2.1.2	Representation of text using an 8-bit coding system e.g. ASCII. *Other representations of a code.*
		2.1.2	Importance of collating sequence in a character-coding system (e.g. ASCII code). Problems with representing international character sets. *The Unicode*
	Computer Logic	2.2.3	Units of storage: bit, byte, kilobyte, megabyte, gigabyte and terabyte. *Storage space of typical media.*
	Logic Circuits	2.2.4	Truth values *(0/1, T/F, on/off).* OR, AND (2-input only) and NOT gates and

			their truth tables. *(Including the Boolean expression for single gates eg. X=A.B, X=A+B and X=$\overline{A}$)* Determining the output of a logic circuit for the 3 mentioned gates. *(Construction of truth table for a logic circuit which may have 2/3 inputs). Truth tables using true/false. (Boolean expression for a circuit is not expected at this stage.)*
	Storage	2.2.6	Main/primary memory – RAM volatile and writeable and ROM non-volatile and non-writeable. Uses of each (eg bootstrap loader in ROM and *running programs in RAM*). Secondary/backing storage devices and media. *Speed difference between main and secondary storage.* Magnetic media (hard disks, tapes, etc). Optical media (CD-ROM, DVD-ROM, etc). Electronic media (pendrives, *etc*). Uses of each, relative capacities (*eg hard disk higher capacity than CD-ROM)*, relative speeds (*eg DVD-ROM is slower than pendrive)* and access modes (direct vs sequential). How a disk drive works – read/write heads, tracks, sectors). *The difference between a backing storage medium and data backup.*
		2.2.6	The disk filing system - storage blocks *(smallest unit of data that can be transferred)*, disk directory, file allocation *(NTFS, FAT)*. Hierarchical directory structure. Access time (typically, short or long only).
1	**COMPUTER APPLICATIONS**		
	Serial and Direct Methods of Access	1.2	Serial and direct access and their suitability of use for certain applications (eg. Serial for payrolls, direct for airline booking reservations).
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	Input devices	2.2.7	Different types of input devices and the application areas for which each is best suited. Qwerty keyboard – alphanumeric and special keys. Pointing devices - mouse, trackball, trackpoint,

			touchpad, light pen and touch screen. Bar code reader, graphics tablet, image (optical) scanner and digital camera. Application of input devices such as OMR, MICR and OCR software, handwriting recognition and pen computing. Joystick, games paddle and audio input (microphone). *Electronic cards (magnetic and chipped).*
	Output devices	2.2.7	Distinction between hard copy and soft copy and between vector devices (eg. plotter) and raster devices (eg. laser printer and screen). Resolution of a raster device (*pixels or dots per unit length*) and how this relates to the print quality and amount of data that needs to be transferred from the computer to the device (and hence speed). LCD projectors. Dot-matrix, inkjet and laser printers *(impact and non-impact)* Plotter. VDU/Monitor. Audio output (loudspeaker).
		2.2.7	How an image is displayed on a CRT, FPD (Flat Panel Display) or LCD. Pixels and colour depth. Palettes. A brief overview of the I/O Subsystem: the speed difference between the CPU/RAM and I/O devices. I/O buffering. Disk caching. Serial vs parallel data transfer.
	Special Purpose I/O	2.2.7	This section is intended to make students aware that people with special needs can still use computers through special devices. Braille printers and keyboards, special purpose keyboards, eye sensor readers and speech recognition.
4	**ALGORITHMIC PROBLEM SOLVING AND PROGRAMMING**		
	Definition and Analysis of Problems	4.1	Analyse the requirements of a problem and create specifications and target for the solution. Specifications should concentrate on inputs and outputs.
	Designing a Solution to the Problem	4.2	The development of algorithms. The use of flow charts for describing an algorithm.
		4.2	The use of structure diagrams and pseudo-code for describing an algorithm

FORM 3 – PRACTICAL COMPONENT		
Topic	**Ref**	**Details**
Basic features of an operating system	2.2.2	Running several applications concurrently in different windows, easy to use graphics interface, use of clipboard to exchange text and graphics data between applications. In particular the integrated use of spreadsheets, word processing and databases (*also graphics programs)*. Managing of files: copying, deleting, renaming, creating folders/directories.
Utility software	1.3	*Utilities that come bundled with OS and as separate packages.* *Function of each utility.* *Format* *Scandisk* *Defragmentation* Antivirus *Compression software (eg. Winzip)*
The web browser	1.7	Simple use of a web browser to access web sites and for searching using a popular search engine. Organising sites in folders. Navigating among sites using the web browser. *Knowledge of:* *Difference between Internet and WWW.* *Logging in* *Web Browser* *Home page* *Urls* *Hypertext* *FTP and HTTP* *Search engines* *Favourites/Bookmarks* *Managing Favourites/Bookmarks* *Security/trusting of sites*
E-mail	1.7	Using an email program to send and receive messages. Saving, printing and deleting a message. Advantages and disadvantages of email compared to the postal system. *Knowledge of:* *User account* *Mailbox – in/out* *Organising mail in folders* *Attachments* *Replying/Forwarding* *Security of mail*

Common application software	1.3	Classical commercial packages – wordprocessing, DBMS, spreadsheet and graphics. The ability to suggest the most suitable software for use in specific environments. The ability to compare and contrast different packages. It is imperative that a demonstration should emphasise the integrated use of these packages and not merely present each one in isolation. Basic skills in the use of the packages mentioned (*and elicited below - DBMS in Form 5)*.and their use under a windowing environment
The Word Processor	1.5	Its advantages and ease of use to produce a simple document such as a letter. Its main features such as fast, easy entry and editing of text, word wrap, margin justifications, centering of text, underlining, indentations, page layout, blocks, find, find/replace, spell check, mail merge. *Additional features include styles, table of contents, indexing and multicolumn layout.*
The Spreadsheet	1.4	Its use to process information. Only simple understanding required, eg. storage of data in cells as a label, value or formula, simple calculations, copying and moving cells, data graphing, printing.
Database Management System	2.1.4	*The difference between database and DBMS.* The organization of data: files (tables), records, fields, items, key fields. *File Specifications.* *Field types to include numeric, text, Boolean (yes/no) and date.* Updating the data base, inserting *(appending)* new records, deleting unwanted records and changing *(editing)* items and fields. Sorting a file according to some criteria eg alphabetically by name. Sorting by more than one field. Forms: Displaying or printing of chosen fields from selected records or from all records. Queries: selecting records under certain conditions. Reports: retrieval of records to view on computer system or print. Relational database: concept of tables and

		relationships (1-to-1 and 1-to-many). Fixed length and variable length records. Importance of speed of response and file structure (serial and direct access) and their applications.
Graphics Packages	1.6	Basic tools to create a simple graphic and its inclusion into text documents.
Definition, analysis, design (implementation and evaluation) of the solution to a problem using the Lego Mindstorms Kit.		*Group work – preferably not more than 3 students per group.* *Brick only: flowcharting and programming with the built-in functions* *Assembled brick (use of sensors + motors): flowcharting and programming with the built-in functions* *NXT- G computer interface: flowcharting and programming concepts through the PC.* *Vide Teacher's Note overleaf.*

FORM 3 NXT MINDSTORMS KIT PRACTICAL TASK
TEACHERS' NOTE

This practical component carries 15% of the Form 3 Annual Examination assessment and consists of a task in which an NXT contraption is programmed using the computer system. It is expected that each student produces his/her own work, even though groups of two/three students working on separate PCs will need to share one kit/contraption. The teacher may consider having different models constructed, so that students could have a wider choice of tasks to choose from.

The task is to be word processed and printed for moderation purposes. The documentation should also include the student's details and the completed marking scheme available at the end of this section. Teachers (and students) may opt to use a presentation software instead of a word processor provided that all previously mentioned requisites are included as hardcopies. The use of the presentation is to be encouraged so that at the very end of this practical component (and if the time frame permits), students may present their tasks to the other groups. Keep in mind that excessive length of the documentation will NOT contribute to a better mark. It is however imperative that the content listed hereunder is included and that it is clearly and visibly indicated.

It is expected that teachers provide students with suggestions and guidance in the choice of the task and to guide them towards problems that:

- Meet their abilities;
- Can be reasonably carried out within the set time frame and with the hardware available.

The teacher may also accept any other task suggested by the students provided it is within the students' capabilities and will be finished within the set time frame.

Teachers are to avoid setting tasks with excessively complex contraptions. It is also imperative that teachers provide students with either a partially-completed or pre-assembled contraption.

CONTENT OF DOCUMENTATION

Problem Definition

This should take the form of a short paragraph clearly stating the aim of the task and any other relevant details to help understand the assigned task.

I/O Device Requirements

This section need only consist of two separate lists, one for the input devices and the other for the output devices that shall be used in the implementation of the assigned task. The student must justify why the particular devices were chosen. The task assigned to the student must include the use of:

- At least two output devices from: Motor, Screen display (on brick) and Loudspeaker (sound from brick), lamp.
- At least two input devices from: Ultrasonic sensor, Sound sensor, Touch sensor and Light sensor.

Algorithm Design

(When the teacher is preparing the student's tasks he/she should ensure that each task shall include the three basic constructs.)

This section consists of a flowchart/s (or pseudocode) of the solution to the assigned task. The flowchart/pseudocode should clearly show the three basic algorithmic constructs, namely, the Sequence, Selection and Repetition. The assigned task may include multiple use (or nesting) of the three constructs.

Program Printout

A screenshot (or File/Print) of the program. The printout should also contain 'inline' documentation using the Comment Tool available in the toolbar. The comments should provide a generic understanding of how the program functions and clearly identify and indicate:

- The sequence, selection and looping constructs;
- The end conditions of loop/s.

Evidence of the Solution

As evidence that the solution to the problem tackled was successful, it is suggested that at least two different photos of the contraption while the program is running be included. It is also suggested that the photos are taken by the students themselves. A short appropriate caption above/beneath each photo explaining the action of the gadget at the instant the photo was taken, should be included.

Evaluation

A short paragraph/s explaining:

- any restrictions/limitations that the student encountered during the design and/or programming stages;
- any possible improvements to the gadget, design and/or program.

FORM 3 NXT MINDSTORMS TASK

MARKING SCHEME

CRITERIA FOR ASSESSMENT	MAX MARK	ACTUAL MARK
Problem Definition	**4**	
I/O Device Requirements • Two input devices and justification • Two output devices and justification	 **3** **3**	
Algorithm Design	**6**	
Program Printout	**6**	
Evidence of Solution	**4**	
Evaluation • Limitations • Improvements	 **2** **2**	
Total/30	**30**	
TOTAL/15 (rounded to nearest integer)	**15**	

Guidelines for awarding marks:

Problem Definition [4]:
A complete and clear aim – 2 marks;
Additional relevant details – 2 marks.

I/O Device requirements [6]:
Input devices:
Mentioning at least two devices – 1 mark;
Justification (1 mark each device) – 2 marks.
Output devices:
Same as for Input devices – 3 marks.

Algorithm Design [6]:
Use of the 3 constructs (1 mark each) – 3 marks;
Correct use of symbols/keywords – 1 mark;
Correct logic – 2 mark.

Program Printout [6]:
Program reflecting the design – 2 marks;
Comments identifying the selection and looping constructs – 2 marks;
Comments explaining other strategic sections of the program – 2 marks

Evidence of Solution [4]:
At least 2 photos (1 mark each) – 2 marks;
Captions explaining photos (1 mark each) – 2 marks.

Evaluation [4]:
Explanation of limitations – 2 marks;
Explanation of improvements – 2 marks.

Student's Name:

_____

Class:

_____

Teacher's Name:

_____

Signature:

_____

Date: _____

Form 4

Learning Outcomes

By the end of Form 4 the learner shall be able to:

- Expand his/her understanding of logic gates to logic circuits and Boolean expressions;
- Understand the concept of a register, its operations and limitations;
- Perform basic binary mathematical functions in 2's complementation;
- Outline the main stages of systems analysis, the important tasks involved at each stage and the relevance of the system life cycle;
- Differentiate between user, technical and program documentation;
- Distinguish between the three types of programming errors: syntax, logical, run-time and methods of detection and correction;
- Demonstrate an understanding of the main components of a CPU and the main memory and the interaction between them;
- Understand the role of the bus system and its relationship to the computer performance;
- Give a brief account of the fetch and execute cycle;
- Distinguish between off-the-shelf, customisable and tailor-made packages and understand software installation;
- Have a notion of the use of computers in the various spheres areas of application including: commercial data processing, scientific uses, communication, education, industry, leisure and home use, office automation, finance and travel;
- Comprehend and modify LeJOS programs;
- Implement solutions to problems using the Java programming language.

FORM 4 – THEORETICAL COMPONENT			
Part	**Topic**	**Ref**	**Detail**
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	Logic circuits	2.2.4	Evaluating a Boolean expression (propositional logic formula) – such as (Ā+B).C given the values of the Boolean variables, by converting to a logic circuit, *i.e. drawing the circuit from the Boolean expression or from the truth table. Construction of a truth table/logic circuit from a textual description of the problem.*
	Number Systems	2.1.1	Concept of a register. Operations of register, complementation, ranges, left and right shifts. *Addition of two binary numbers.* Numerical overflow.
		2.1.1	2's complement representation. Binary addition and subtraction (by complementation followed by addition) in 2's complement.
	Coding Systems	2.1.2	Range of possible symbols that can be represented by a given number of bits.
3	**COMPUTER SYSTEMS**		
	Systems Analysis	3.3	The process of analyzing a system with the view to computerization: 1. Project selection and feasibility study, 2. Present system study and analysis, 3. Design of new oolean able system, 4. Programming and documentation, 5. Implementation and changeover methods, 6. Control and review 7. System maintenance
4	**ALGORITHMIC PROBLEM SOLVING AND PROGRAMMING**		
	Documentation	4.5	The difference between user, technical and program documentation. Describing and giving examples of essential features found in user documentation accompanying software packages.

	Testing	4.4	Types of errors: syntax, logical, run-time (*execution)*. Ability to distinguish between them and give examples. Appropriate methods of detection and correction. Use of suitable test data to check the performance of a program. Output of intermediate results. Dry running. Program tracing.
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	The CPU	2.2.5	The processor and main memory. The main components of the processor – control unit, ALU. Registers: Accumulator, Program counter (PC), Instruction register (IR). Main memory and memory addresses. The address bus and how its width relates to the size of the address space. The data bus. The computer's word size (or word length). Control bus (read/write line only). Concept of a stored program *as a set of instructions*. Brief account of fetch and execute cycle. *[Refer to Appendix 2].* Processor speed – cycles per second expressed in Hz, KHz, MHz and GHz. Units of time measurements: milli, micro, nano seconds.
3	**COMPUTER SYSTEMS**		
	Programming and Application Packages	3.1	Off-the-shelf, customistable and tailor-made packages. Advantages and disadvantages of each. *System and application software Installation, on-line help.*
5	**ICT IN SOCIETY**		
	Areas of Computer Applications	5.1	The range of applications should include: Commercial data processing: e.g. Stock control, reservations, administration, POS systems. Technical, mathematical and scientific uses: e.g. Medical diagnosis, CAD, simulation,

			weather forecasting. Communication and information systems: Internet and WWW, electronic mail, e-government. In industry: computer process control: e.g. industrial processes, robotics, gas and oil exploration, monitoring and using energy, CAD-CAM Educational uses: e.g. CAL, inter-school projects using Internet, school administration. Leisure and home uses: e.g. games, microprocessor-controlled home appliances, air-conditioning, security systems. Office automation: word processing, database systems, spreadsheets, creation and use of graphics presentation software, personal organizers and schedulers. Finances: shops, banks, EFT, stock control, supermarkets, stock exchange, insurance, e-commerce. Travel: air traffic control, navigation (e.g. GPS), 'intelligent' cars, space travel. In the community: police, health, schools, ecological interests, libraries, supermarkets, teleshopping.

FORM 4 – PRACTICAL COMPONENT		
Topic	**Ref**	**Details**
LEGO MINDSTORMS KIT – LEJOS		Candidates are ONLY required to ***read*** and ***modify*** programs in LeJOS and they are NOT required to ***write*** programs in LeJOS. The following features should be covered and can be examinable in LeJOS: • Using the following methods provided by the ***Motor*** class: ***backward(), forward(), setSpeed(int speed), stop(), regulateSpeed(oolean yes)*** • Using the method ***sleep()*** provided by the ***Timer*** class • Using the ultrasonic sensor by making use of the constructor ***UltrasonicSensor(SensorPort port)*** to create an instance (object) and using the call ***getDistance()*** to obtain the required distance. • The use of the keyword '***new***' to create a new object. N.B. Use of the other sensors is advisable but is not examinable.

FEATURES OF PROGRAMMING LANGUAGE – JAVA	4.3	(The sequence of the features does not necessarily dictate the order in which they are to be taught). 1. Valid meaningful identifiers, enforcing code convention rules (See Appendix 3), case sensitivity. 2. Variable and constant declarations 3. Data types (primitive), compatibility and type casting • byte, short, int, float, char, oolean. 4. The type String can be used for alphanumeric data. No knowledge of String class methods is required. 5. Initialisation and assignment operator – equal sign (=) 6. Comments • // (single line) and /* (multi-line) 7. Arithmetic operators, precedence and expressions • +, -, *, /, %, ++, --, +=, -=, /= , %= 8. Printing and formatting of text-based output, escape characters: • print(), println(), \', \n, \t (tutors may use printf () method for formatting purposes especially decimal numbers) 9. Inputting data from the keyboard through a third party class (the keyboard class).

		10. Logical operators, simple and compound logical expressions • !(unary not), && (and), || (or), == (equal to), != (not equal to), > (greater than), >= (greater or equal), < (smaller than), <= (smaller or equal). 11. Loops (including nested loops): • while, do while and for 12. Conditional transfer (including at least 2 level nesting): • if, if – else, switch 13. Math class – the Math class should be explained as one of the libraries/classes found in the JDK, and limited to the following methods: • abs(), pow(), sqrt(), random(), round(), ceil(), floor(). 14. Declaration and creation of single-dimensional arrays of primitive data types. 15. Objects – Students create their own classes containing simple void methods and create instances through the use of the keyword 'new'. *Refer to Appendix 3 for code convention rules.* *Syntax, logic and runtime errors.*
		Declaring and calling methods with and without arguments and simple methods which return a value.

Sample Java SEC Examination questions can be found at:

- http://www.um.edu.mt/matsec/resources/resources

FORM 4 ANNUAL – JAVA PROGRAMMING COURSEWORK

This Coursework carries 15% of the annual examination mark. This exercise is to be fully word processed and to include the items outlined below.

Problem Definition

- A brief description of the application developed and a possible context for its use.

Solution of the problem

- A structured algorithm described by means of a suitable diagrammatic methodology or pseudo-code.

- A hardcopy listing of the program. The program should include the following constructs:
    - inline documentation
    - input and output statements
    - assignment expressions
    - decision construct *
    - looping construct *
    - implementation of array
    - simple objects and methods
    - any special design features

* The student is free to choose the decision and looping constructs best suited for his purpose.

Notes:

This exercise may or may not be later related to the SEC coursework.

It is expected that teachers give guidance and suggestions to all candidates in the choice of programming tasks, relating to their ability as well as to hardware and time constraints.

While students may be encouraged to come up with their own ideas for the coursework or their own variations on suggested programs, the following are just some examples of programming tasks that can be set:

Class marks

An application designed to handle basic class mark management procedures. These could possibly include all or some of the following:

- Entry of marks in an array.
- Basic data processing:
  - Generation of statistics: average mark, lowest mark, highest mark, marks above average, marks below average.
  - Generation of grades.
  - Generation of simple histogram showing grade distribution.

Educational Quiz

An application that sets the user an educational quiz. The system may display a series of questions and read the user's replies one by one. An array can be used to hold questions and answers. Marks could be awarded for each correct answer. The program may give the user some or all of the following options:

- The option to choose a quiz topic.
- The option to demonstrate the correct answers to the test once the user has tried the test.
- The option for the teacher to enter a password to then be allowed to edit the quiz questions.

Telephone directory

An application that provides basic telephone directory functions. Names, addresses and their related telephone numbers can be held in an array. The program may give the user some or all of the following options:

- The option to search for a name and output the telephone number, town or address associated with it.
- The option to output the whole directory
- The option to search details by a particular town.

Any **other program** can be developed as long as candidates get their teacher's approval to ensure that it is at the correct level.

FORM 4 ANNUAL
PROGRAMMING COURSEWORK MARKING SCHEME

Criteria for Assessment	**Maximum Mark**	**Actual Mark**
Problem Definition:		
Description of the scope of the problem to be tackled	**2**	
Solution of the problem:		
Algorithm (flowchart or pseudocode)	**4**	`
Computer listing of program involving		
o Declaration of user-defined classes	**2**	
o Creation of objects from defined classes	**2**	
o Simple methods	**3**	
o Input and output statements	**2**	
o Assignment expressions	**2**	
o Decision construct	**3**	
o Looping construct	**3**	
o Implementation of an array	**3**	
o Inline documentation	**2**	
o Special design features	**2**	
TOTAL	**30**	
TOTAL/15 (rounded to nearest integer)	**15**	

Guidelines for awarding marks:

Problem Definition

Description of the scope of the problem to be tackled [2]

A complete and clear aim and any additional relevant details

Solution of the problem

Algorithm (flowchart or pseudocode) [4]

Award full 4 marks for a complete, readable, language-independent and easy-to-follow algorithm

Computer listing [24]

As per marking scheme above

Special design features: Marks are to be awarded for an adequate implementation of any special features included in the program (good user interface, use of advanced constructs and algorithms, etc).

Student's Name:

_____

Class:

_____

Teacher's Name:

_____

Signature:

_____

Date:

_____

Form 5

Learning Outcomes

By the end of Form 5 the learner shall be able to:

- Distinguish between the different language levels;
- Differentiate between source code and executable code and the types of translators and their uses;
- Show an understanding of the preparation of data for input, the related transcription errors and methods of detection of errors;
- Understand and modify simple assembly language programs and have the notion of immediate, direct and symbolic addressing;
- Outline the role of system software and, in particular, of the Operating System in resource management;
- Differentiate between CLI and GUI and their typical uses;
- Differentiate between different types of Operating Systems and their uses;
- Differentiate between a general-purpose and a dedicated computer and the relationship between embedded and dedicated systems, identifying special I/O devices;
- Differentiate between the types of networks: LAN, MAN, WAN and WLAN and identifying the advantages and disadvantages of each;
- Differentiate between server and client machines on a network;
- Show an understanding of software licensing considerations;
- Show an appreciation for the positive and negative aspects of computerization on different social spheres;
- Identify tasks performed by the following people in the IT department: Systems Analyst/Designer, Programmer, I.T. Trainer, Data Entry Clerk, Web Master, Computer technician, Computer engineer;
- Use a Desktop Publisher to create a simple publication and a web authoring software to create a simple website;
- Make use of basic features of the following: Graphics and image editing software, Multimedia, CAD, Organizers and Schedulers, Games;

FORM 5 – THEORETICAL COMPONENT			
Part	**Topic**	**Ref**	**Detail**
4	**ALGORITHMIC PROBLEM SOLVING AND PROGRAMMING**		
	Translation of High-Level Languages	4.6	Languages of a higher level than assembly language. The need for translators. Language translation as a transformation which preserves the semantics of a program – hence the high-level statement may result in many low-level instructions
	Language translators	4.7	The difference between compilers, interpreters and assemblers The relative advantages and disadvantages of each. Source code and executable code. Error messages.
		4.7	4GLs. Software portability.
3	**COMPUTER SYSTEMS**		
	Programming and Application Packages	3.1	Awareness of the importance of choice of programming language in developing application software. Awareness of 4th generation languages – demonstration of designing a database Software licensing considerations.
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	Preparation of Data for Inputting into the Computer System	2.1.3	Data capture forms (very simple). Preparation and transcription of data with their related errors *(transposition, omission, substitution)*. Solution through data verification and validation; check digits; range check.
	The CPU	2.2.5	Concept of language levels (machine code being at a lower level than assembly language). The instruction set as a means of controlling the CPU's circuitry. Function codes (opcodes) and operands. Concept of a machine code program as a set of instructions.

		2.2.5	Typical machine code instructions - load, store and process instructions. Using mnemonics to represent machine instructions. Immediate, direct and symbolic addressing. Conditional and unconditional branches. Understanding and modifying simple assembly language programs. *[Refer to Appendix 1]*
	The Operating System	2.2.8	*Interfaces - CLI and GUI.* System software as a layer between the user/application and the hardware. Resource management functions; management of files, memory, CPU, I/O. *Typical examples of each function.*
3	**COMPUTER SYSTEMS**		
	Types of Operating Systems	3.5	Real time *[Refer to Appendix 2]*, batch, time sharing (on-line) use. Their meaning – differences shown by examples. The suitability of each operating mode for a particular application. Common types of operating systems – single-user, multi-user, networked, single programming, multi programming.
2	**COMPUTER ARCHITECTURE AND DATA REPRESENTATION**		
	Dedicated Computer Systems	2.2.9	Difference between a general-purpose and a dedicated computer. Embedded and process control systems. Computerised appliances as examples of dedicated systems – VCR, auto pilot, optical recorder/player, mobile phones, GPS, etc. Specialised I/O devices (sensors, buttons, LCD). Software for dedicated computer systems.
3	**COMPUTER SYSTEMS**		
	Networks	3.4	Networking: LAN, MAN, WAN and WLAN as a variation of a LAN. Advantages of a LAN over a number of standalone PCs – sharing of hardware and software resources, ease of

			management/control by the system administrator. The use of modems to interface computers with telecommunications networks (telephone cable, optic fibre, microwave, satellite links). Computer communications over WANs; email, WWW, Video Conferencing. *(Other services over WANs may be included eg chatting, etc)*
		3.4	Server and client machines. The problem of bandwidth at a general level.
5	**ICT IN SOCIETY**		
	Data Security and Privacy	5.3	Need of data security and integrity of data. Backups (the generations of files: grandfather, father, son files). *Parity checking.* Physical security and software safeguards. Malta Data Protection Act – 2001. The provisions and implications of the Act for the various sectors and citizens. Software piracy and copyright. Ethical and legal issues. Hardware and software procedures which deter piracy - serial numbers and activation keys, hardware keys (dongles). Software registration.
		5.3	Access rights. Privacy on multi-user/network systems.
5	**ICT IN SOCIETY**		
	Effects of Computer-based Systems on Individuals, Organisations and Society	5.2	Positive and negative effects of computerisation, eg. satisfaction and efficiency at work, effects of computer games on youngsters, health hazards from working long hours at a computer, opportunity for crime.
	Multimedia	5.4	Brief overview of capabilities and trends. Future perspectives (home office, access to public and institutional databases, libraries, high-quality sound and pictorial data representation). Basic hardware and software requirements and costs.

3	**COMPUTER SYSTEMS**		
	Roles Related to an IT Environment	3.2	Knowledge of the existence of a wider range of tasks and responsibilities and hence the need to share them. The ability to outline the duties of: Systems Analyst/Designer Programmer I.T. Trainer Data Entry Clerk Web Master Computer technician Computer engineer

FORM 5 – PRACTICAL COMPONENT		
Topic	**Ref**	**Details**
SEC Programming Coursework		*Programming exercise.* *The coursework must be presented to the teacher for correction by the end of the calendar year.* *Vide Teachers' note below*
Desktop Publisher	1.5	Main DTP features such as automatic table of contents and index creation, multi-column documents, tables, frames, embedded graphic objects, etc.
Creating Web pages	1.7	Using a web authoring program to edit/create a simple web page that includes text, graphics and buttons for linking to other sites.
Other application software		*Teachers may find time to demonstrate:* *Graphics and image editing* *Multimedia* *CAD* *Organizers and Schedulers* *Games*

FORM 5 ANNUAL – JAVA PROGRAMMING COURSEWORK FOR SEC EXAMINATION

TEACHERS' NOTE

This practical component completes the Programming exercise as requested by the MATSEC board. Also, it carries 15% of the Form 5 Annual Examination assessment. Each student has to produce his/her own work.

Marks are awarded for the following sections in the coursework.
Twenty six marks out of the 30 marks are allotted for:

- Definition of the Problem
- Solution of the Problem
- Running the program
- User instructions
- Comments and conclusions

The other 4 marks are to be awarded for the overall layout and presentation of the coursework. More details may be sought from the official SEC syllabus.

This global practical mark for the SEC examination shall also be used to satisfy the 15% set aside for the practical component of the Form 5 annual examination mark. Obviously the mark has to be divided by 2. With this in mind, it is imperative that the practical component be finalised by the end of the calendar year so that the teacher has ample time to finalise his/her corrections before the commencement of the Form 5 annual examination session in February.

Moderation of coursework at Form 5 may be exercised by us and/or MATSEC personnel.

SECONDARY EDUCATION CERTIFICATE IN COMPUTING
TEACHER –ASSESSED PRACTICAL SCORE - COURSEWORK MARKING SCHEME

IMPORTANT: For moderation purposes by MATSEC, this completed sheet is to be available together with the coursework exercise.

Criteria for Assessment	**Maximum Mark**	**Actual Mark**
A: PROGRAMMING		
Definition of the problem:		
Description of the scope of the problem to be tackled	**2**	
Statement of the results required	**2**	
Details of the input information required	**2**	
Solution of the problem:		
Algorithm (flowchart or pseudocode)	**4**	`
Computer listing of program	**4**	
Details of any special design features	**4**	
Running the program		
Evidence that the solution works	**2**	
Plan of test data	**2**	
User instructions:		
Loading and operating the program	**2**	
Comments and conclusion:		
Limitations and improvements	**2**	
TOTAL MAXIMUM FOR PROGRAMMING	**26**	
B: LAYOUT AND PRESENTATION		
Clear indication of each section mentioned above	**1**	
Use of headers and footers (to include page numbers)	**1**	
Use of tables	**1**	
Use of bulleted and/or numbered lists	**1**	
TOTAL MAXIMUM FOR LAYOUT/PRESENTATION	**4**	
TOTAL	**30**	

Guidelines for Awarding Marks:

- Sections with a maximum mark of 2:
  Award marks as follows
  - 0 – not attempted;
  - 1 – partially presented;
  - 2 – fully correct response.
- Algorithm: award 4 marks for a complete, readable, understandable, language-independent and easy-to-follow algorithm
- Computer listing: award 4 marks for programs showing meaningful identifiers, correct indentation, use of correct coding rules and inclusion of comments
- Special design features: award 2 marks for an adequate description of any special features included in the program (good user interface, use of advanced constructs and algorithms, etc). The other 2 marks are to be awarded for a description (and the eventual use) of simple objects.

Student's Name in blocks:

Tutor's Information:

Name in blocks:

Signature:

Date:

APPENDIX 1

ASSEMBLY LANGUAGE

(Applicable only to Track 3)

The list of limited assembly language instructions given below will be assumed. The operand can either be a memory address (direct addressing), actual data (immediate addressing) or a symbolic address. The symbol # will be used to specify immediate addressing:

```
Eg    ADD  #12     – immediate addressing
      ADD total    – symbolic addressing
      ADD 36       – direct addressing
```

For the purpose of this section the size and type of accumulator is 8-bit unsigned. Comments are introduced by a semicolon (;).

Data transfer instructions:

```
LDA  x    ; Load accumulator A with x
STA  x    ; Store contents of accumulator A in x
```

Arithmetic, Logical and Shift instructions:

```
ADD  x    ; Add contents of accumulator A with x
SUB  x    ; Subtract contents of x from accumulator A (A minus x)
MUL  x    ; Multiply contents of accumulator A with x
DIV  x    ; Divide contents of accumulator A by x

AND  x    ; Logical AND the contents of accumulator A with x (bitwise operation)
ORA  x    ; Logical OR the contents of accumulator A with x (bitwise operation)
NOT       ; Logical NOT the contents of accumulator A (no operand).

SHL       ; Logical shift contents of accumulator A to the left, introducing a 0 in the
            vacated bit (no operand and bitwise operation).
SHR       ;Logical shift contents of accumulator A to the right, introducing a 0 in the
            vacated bit (no operand and bitwise operation).
```

Transfer of control instructions:

```
JMP  x    ; Jump to the instruction pointed to by the label x (unconditional jump)
JZE  x    ; Jump to the instruction pointed to by the label x if contents of accumulator is 0
              (zero)
JNZ  x    ; Jump to the instruction pointed to by the label x if contents of accumulator is not
              0 (zero)
```

Other instructions:

```
HLT       ; End of program
```

The operation performed by each instruction is to be provided as a comment in examination papers.

APPENDIX 2

REAL-TIME OS

Consider time critical **real-time** operating systems as applied in aircraft control systems, nuclear power stations, missile guidance systems, anti-missile attack systems, and other process control systems with a critical response time. Do not include airline booking systems as examples of time critical real-time operating systems.

Characteristics of real-time system:

a) support application programs which are non-sequential in nature;

b) deal with events occurring concurrently

c) process and produce a response within a guaranteed specified time interval;

d) safely-critical systems with hardware redundancy

THE FETCH EXECUTE CYCLE

Only a general account of the fetch execute cycle is required, consisting of the following steps:

1. Control unit fetches the opcode from the memory location indicated by the Program Counter.
2. Control unit places opcode in Instruction Register.
3. Control unit fetches any required operand.
4. Control unit increments the Program Counter to point to the next instruction.
5. Control unit activates necessary circuits to execute the instruction.
6. Go back to step 1.

APPENDIX 3

Java Code Convention Rules

The enforcing of code conventions make programs more understandable by making them easier to read. They can also give information about the function of the identifier; for example, whether it is a constant or a class. This can be helpful in understanding the code.

Identifier Type	Rules for Naming	Examples
Classes	Class names should be nouns in singular, in mixed case with the first letter of each internal word capitalized. Keep class names simple and descriptive.	class Raster; class ImageSprite;

Methods	Methods should be verbs, in mixed case with the first letter lowercase, with the first letter of each internal word capitalized.	run();
		runFast();
		selectFromMenu();

Variables	Variable names should be short yet meaningful. The choice of a variable name should indicate to the casual observer the intent of its use. One-character variable names should be avoided.	int num; char letter; float myWidth;
Constants	The names of variables declared class constants should be all uppercase with words separated by underscores ("_").	final int MIN_WIDTH = 4; final int SPEED_LIMIT = 60;

Code block – Opening first curly bracket same line as construct and closing on a separate line.

Example:
```
if (     ) {

} else {
}
```