L-Università ta' Malta

MATSEC
Examinations Board

SEC 09 Syllabus

Computing

2026

Table of Contents

Introduction

This syllabus is based on the curriculum principles outlined in *The National Curriculum Framework for All* (NCF) which was translated into law in 2012. It is designed using the *Learning Outcomes Framework* that identify what students should know and be able to achieve by the end of their compulsory education.

As a learning outcomes-based syllabus, it addresses the holistic development of all learners and advocates a quality education for all as part of a coherent strategy for lifelong learning. It ensures that all children can obtain the necessary skills and attitudes to be future active citizens and to succeed at work and in society irrespective of socio-economic, cultural, racial, ethnic, religious, gender and sexual status. This syllabus provides equitable opportunities for all learners to achieve educational outcomes at the end of their schooling which will enable them to participate in lifelong and adult learning, reduce the high incidence of early school leaving and ensure that all learners attain key twenty-first century competences.

This programme also embeds learning outcomes related to cross-curricular themes, namely digital literacy; diversity; entrepreneurship; creativity and innovation; sustainable development; learning to learn and cooperative learning and literacy. In this way, students will be fully equipped with the skills, knowledge, attitudes and values needed to further learning, work, life and citizenship.

What is Computing?

Computing is an ideal foundation course for further study in Computer Science. Understanding the principles of digital devices, problem solving, and software development provide learners with the underpinning knowledge required for many other subjects in science, engineering and the IT sphere. The skills learnt can also be used in everyday life.

What does a study of Computing entail?

In Computing, candidates take a hands-on approach to technology. The aim is in fact to help candidates to reflect on the technology they use every day and encouraging them to think about how and why it works, and how and why it can be better.

In Computing, candidates are encouraged to focus on real world problem-solving. In fact, a key component of Computing is problem-solving through programming, where candidates get the opportunity to program dedicated systems and create simple applications of their own. This is in fact a source of great satisfaction to the candidates.

Computing provides the candidates with skills required in further education, equipping them for the ever-increasing new jobs in the tech market.

How is Computing related to candidates' lives, to Malta, and/or to the world?

The aim is to spark computational thinking and empower candidates with the twenty first century skills that will let them start preparing today for the problems we will only meet tomorrow, and to make sure they enjoy the opportunity.

The IT sphere is vast and an important part of various aspects of society. Computing will empower students to be able to choose an area they are interested in and continue growing.

The aspirational programme learning outcomes for this subject are:

At the end of the programme, I can:

1. develop an understanding of the components, the architecture and the organisation of a digital device;
2. develop an understanding of the problem-solving process;
3. acquire skills to create algorithms to solve problems;
4. acquire necessary skills to solve computer-based problems using high-level programming language.
5. develop an appreciation for the characteristics of operating systems and their applications;
6. develop an understanding of how computer networks can be used to connect computers together;
7. develop a range of cognitive skills, including critical thinking skills.

List of Subject Foci

1. Introduction to Digital Devices
2. Principles of Computing
3. Machine Logic
4. Databases
5. Fundamentals of Networking
6. Problem Solving, Algorithms and Physical Computing
7. Programming Languages and Fundamentals to Program Development

List of Learning Outcomes

At the end of the programme, I can:

LO 1. identify the function of the main components in a digital device to process both digital and analogue data.

LO 2. represent data using different number systems and perform basic binary arithmetic operations.

LO 3. describe the use and the quality of input and output devices.

LO 4. distinguish between memory and storage in digital devices and describe the implications of the different technologies used.

LO 5. distinguish and identify the importance of different types of software and software licenses.

LO 6. assess the suitability of different CPU specifications for different scenarios.

LO 7. describe the role and the basic functions of an Operating System in different digital systems.

LO 8. understand principles of machine logic in general and produce logic circuits, truth tables & Boolean expressions using the NOT, AND, and OR gates.

LO 9. show an understanding of a relational database, its structure and use.

LO 10. outline networking concepts of how devices communicate.

LO 11. produce algorithms, by applying problem solving concepts, and develop simple dedicated systems using boards powered by microcontrollers or System on Chip (SoC) technology.

LO 12. distinguish between low and high-level languages.

LO 13. develop programs using Python programming language that includes textual and graphical interfaces.

Programme Level Descriptors

This syllabus sets out the content and assessment arrangements for the award of Secondary Education Certificate in Computing at Level 1, 2 or 3. First teaching of this programme begins in September 2022. First award certificates will be issued in 2025.

The following levels refer to the qualification levels that can be obtained by candidates sitting for SEC examinations. These are generic statements that describe the depth and complexity of each level of study required to achieve an award at Level 1, 2 or 3 in Computing. (Level 1 being the lowest and level 3 the highest).

Level 1: At the end of the programme the candidate will have obtained basic knowledge, skills and competences in the subject such as basic repetitive communication skills and the ability to follow basic, simple instructions to complete tasks. Support is embedded within the task.

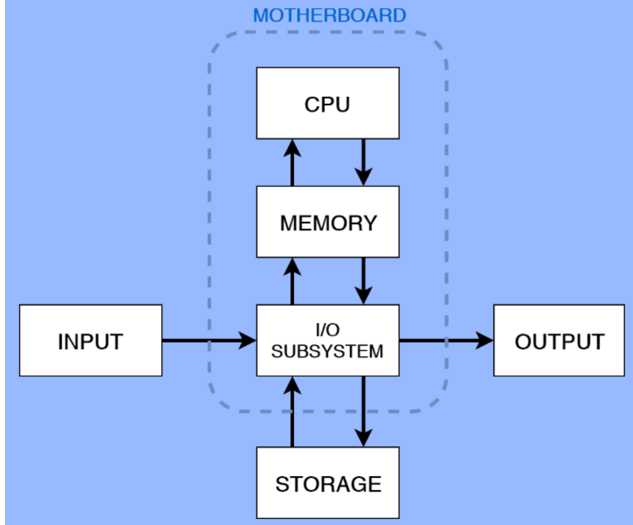
Level 2: At the end of the programme the candidate will have obtained good knowledge, skills and competence in the subject such as the interpretation of given information and ideas. The candidate will have developed the ability to carry out complex tasks. Limited support may be embedded within the task.

Level 3: At the end of the programme the candidate will autonomously apply knowledge and skills to a variety of complex tasks. Candidates will utilise critical thinking skills to analyse, evaluate and reflect upon their own work and that of others. Problem solving tasks may be part of the assessment process.

Learning Outcomes and Assessment Criteria

Subject Focus:	**Introduction to Digital Devices**
Learning Outcome 1: **(Paper II)**	**At the end of the programme, I can identify the function of the main components in a digital device to process both digital and analogue data.**

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
1.1a Define the components/devices in a computer system. *Limited to motherboard, CPU, RAM, and input / output devices and storage devices.* *Note: Each component is limited to the following characteristics:* ▪ *Motherboard: Component that assembles all components together;* ▪ *CPU: processes data and returns the result;* ▪ *RAM: Working memory, Volatility, Read & Write access;* ▪ *Input / Output Devices: Inputting data to device and outputs information to the user;* ▪ *Storage Devices: Components that permanently store data.*		
1.1b Identify the components/devices in a computer system. *Limited to components in 1.1a according to the following hardware diagram:*	1.2b Explain the data flow between the main components/devices in a computer system. *Example: CPU cannot process data directly from storage.*	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
 MOTHERBOARD CPU MEMORY INPUT I/O SUBSYSTEM OUTPUT STORAGE		
1.1c Define data and/or information.	1.2c Distinguish between data and information.	
1.1d Identify examples of data and information according to a given scenario.	1.2d Recommend examples of data and information according to a given scenario.	1.3d Justify the difference between data and information in a given scenario. *Example: Data coming from sensors and interpreted by software to produce graphs and data coming from record logs and the identification of trends.*
1.1e Define binary units of measurement. *Including but not limited to: bits, Bytes, Kilobytes, Megabytes, Gigabytes & Terabytes.* *Note: 1bit = 0 or 1; 8bits = 1 Byte; 1024 Bytes = 1 Kilobyte, etc.*	1.2e Justify why digital devices process data in binary. *Note: binary is representing two-state electrical pulses: high and low.*	1.3e Solve problems related to binary units of measurement.

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
1.1f Define analogue and/or digital data. *Limited to continuous (wave) and discrete values.*	1.2f Distinguish between analogue and digital data.	
1.1g Define analogue-to-digital (ADC) and/or digital-to-analogue (DAC) conversions.	1.2g Explain the need for ADC and DAC conversions.	1.3g Evaluate the implications of the sampling level used during ADC. *Limited to implications of cost, data quality and portability.*
1.1h Define a microcontroller. *Limited to CPU, ROM, RAM and storage components in one chip.*		
1.1i Identify devices powered by a microcontroller. *Examples: programmable thermostats, household appliances, industrial instruments,, toys, microcontroller-based platforms, such as Arduino etc.*	1.2i List devices that are powered by a microcontroller.	
1.1j Define System on Chip (SoC) technology. *Limited to an entire system (CPU, ROM, RAM, storage and other components, such as GPU, modem and WIFI card) into one chip.*		
1.1k Identify devices using SoC technology. *Examples: smartphones, tablets, single-board computers, such as Raspberry Pi boards, wearable devices, digital cameras, wireless routers, smart TVs, etc.*	1.2k List devices that use SoC technology.	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	1.2l Distinguish between microcontroller and SoC. *Limited to:* ▪ *Microcontroller used in embedded systems that can only perform a dedicated task, or several tasks related to a single use;* ▪ *SoC used for more powerful applications and processing capability of a computer and can do all the tasks, such as running an OS*	1.3l Justify the appropriateness of SoC and/or microcontroller technology in a given scenario. *Limited to device portability, power efficiency, lack of adaptability, and low production costs but initial manufacturing costs.*

Subject Focus:	**Principles of Computing**
Learning Outcome 2: (Paper II)	**At the end of the programme, I can represent data using different number systems and perform basic binary arithmetic operations.**

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
2.1a Recognise different number systems. *Limited to binary, decimal and hexadecimal.* *Note: unit used is base 2, 10 and 16.*	2.2a Distinguish between the use of different number systems. *Limited to binary, decimal and hexadecimal.*	2.3a Solve problems related to different number systems in specific scenarios. *Note: candidates are expected to use positional notations to solve problems in a scenario.* *Example: the use of 62 base number system for URL-Shortener.*
	2.2b Outline the use of a bit pattern according to a given scenario.	
	2.2c Convert binary numbers into decimal and vice versa.	
	2.2d Convert hexadecimal numbers into binary and vice versa.	
	2.2e Convert hexadecimal numbers into decimal and vice versa.	
	2.2f Represent positive and negative numbers in binary using two’s complementation method.	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	2.2g Carry out binary addition and/or subtraction. *Limited to:* ▪ *up to three unsigned numbers for addition;* ▪ *the use of the two's complementation method and up to two numbers for subtraction.*	2.3g Justify the use of two's complement representation. *Note:* ▪ *eliminate the problem with one's complement, in that 0 has a double representation;* ▪ *flexibility of the circuit to process both addition and subtraction.*
2.1h Relate arithmetic bit shift operations with multiplication and division operations.	2.2h Carry out binary multiplication and division. *Limited to: the use of the bit shifting operations and unsigned whole numbers.*	
	2.2i Solve problems related to range of values in a given register. *Limited to unsigned and two's complement registers.*	2.3i Justify the occurrence of numerical overflow.
2.1j Define standard character coding systems. *Limited to the need to provide a universal mode of representing characters on different platforms.*	2.2j Identify the need of different standard character coding systems. *Limited to ASCII (8 bits) and UNICODE (16 bits).*	

Subject Focus:	**Principles of Computing**
Learning Outcome 3: (Paper II)	**At the end of the programme, I can describe the use and the quality of input and output devices.**

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
3.1a Define the concept of the input of data into digital devices.		
3.1b Define the concept of the output of information from digital devices.		
3.1c Define peripheral devices.	3.2c Categorise components in digital devices as being input, output or both. *Note: Description of specialised devices must be given in exam questions.* *Examples: smartphone device has the following components: speaker, camera, flash, microphone, screen, satellite receiver, accelerometer sensor, etc.*	3.3c Evaluate the use of different components in a specific scenario.
	3.2d List different input and/or output and/or input/output components in a specific scenario.	3.3d Recommend different component/s according to a specific scenario.
		3.3e Justify the choice of different component/s according to a specific scenario.
3.1f Define serial and/or parallel data transfer between peripheral devices and digital systems.	3.2f Distinguish between the concepts of serial and parallel data transfer.	3.3f Explain advantages and/or disadvantages of serial and/or parallel data transfer in a given scenario. *Limited to scenarios in terms of data transfer to/from peripheral devices and digital systems.* *Example: data transfer to/from peripheral devices have a problem with data synchronisation.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
3.1g Identify the different characteristics which determine output quality. *Characteristics limited to:* *1) visual resolution, bit depth, raw and compressed images, and frames per second;* *2) sound sampling including concepts of sound compression.* *Examples: RAW/TIFF image is uncompressed and JPG and PNG image is compressed. WAV sound file in uncompressed and Mp3 are compressed.*	3.2g Explain the different characteristics which determine output quality.	3.3g Evaluate the suitability of resources based on the characteristics which determine output quality in a given scenario. *Examples of resources may include: images, audio files, video clips, animations, etc.*
	3.2h Find solutions that determine the characteristics of output quality.	3.3h Justify the suitability of resources based on the characteristics which determine output quality for a given scenario. *Characteristics limited to those listed in 3.1g.* *Examples of resources as listed in 3.3g.*

Subject Focus:	**Principles of Computing**
Learning Outcome 4: (Paper II)	**At the end of the program, I can distinguish between memory and storage in digital devices and describe the implications of the different technologies used.**

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
4.1a Define primary and/or secondary storage. *Note:* ▪ *Primary storage refers to memory components (RAM, ROM and cache);* ▪ *Secondary storage refers to storage devices such as Hard Disk drive or flash drive.*		
4.1b Define the memory components used in digital devices. *Limited to byte-addressable RAM and/or ROM and/or cache.*	4.2b List different memory components used in digital devices.	
4.1c Identify memory components used in digital devices. *Limited to as in 4.1b.*	4.2c Distinguish between memory components used in digital devices in terms of volatility, speed and use. *Note: differences in both general-purpose systems, such as a PC, laptop or smartphone, as well as in dedicated systems, such as a washing machine, air conditioner, etc.* *Example: ROM in a PC stores the booting up instructions (firmware/BIOS) while ROM in a toy-car stores: 1) start-up routines when the car is switched on; 2) factory settings, such as remote-control frequencies; and 3) set-routines (instructions), such as how the buttons on the remote-control signals different motors/servos in the car.*	4.3c Justify the choice of memory components for specific scenarios. *Example: 16GB RAM is suitable for a gamer PC but not required for an office PC OR 4MB Cache is suitable for an Office PC but an automated traffic light system which is powered by a microcontroller does not require 4MB of cache.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
4.1d Identify devices that are used for secondary storage. *Limited to: magnetic hard disk drive, optical media, SSD, flash drive and flash memory.*	4.2d Distinguish between different storage devices in terms of speed, cost and portability. *Example: a Solid-State-Drive (SSD) is faster than a magnetic hard disk drive but more expensive.*	
4.1e List devices that are used for secondary storage. *Limited to devices listed in 4.1d*	4.2e Recommend storage media for specific scenarios. *Example: Blu-Ray used to store 4K Movie OR flash memory as expandable storage for smartphone, etc.*	4.3e Justify the choice of storage media for specific scenarios.
	4.2f Distinguish between memory components and storage components. *Limited to their use.*	
4.1g Define cloud storage. *Note: in terms of a storage medium making use of a third-party infrastructure as a service and accessed remotely.*	4.2g Describe advantages and disadvantages of cloud storage. *Note: Advantages and disadvantages should be justified in terms of cost, security, access to data, and data sharing issues.*	4.3g Justify the use of cloud storage in specific scenarios. *Example: A doctor has the possibility of accessing a patient's record from his/her personal laptop if a patient comes to the clinic, or smartphone if doing a house visit. Would you use cloud storage? Explain?*

Subject Focus:	**Principles of Computing**
Learning Outcome 5: **(Paper II)**	**At the end of the programme, I can distinguish and identify the importance of different types of software and software licences.**

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
5.1a Define the different types of software. *Limited to: System software, application software, off-the-shelf software and tailor-made software (Bespoke).*		
5.1b Identify the different types of software. *Limited to: system software, application software, off-the-shelf software and tailor-made software (bespoke).*	5.2b Distinguish between the different types of software.	5.3b Justify the type of software chosen for a given scenario.
5.1c Define the concept of a software license.		
5.1d Define the software licenses. *Limited to: freeware, shareware, single-user, site, and open-source licenses.* *Note: this syllabus refers to the licenses as follows:* ▪ *Freeware: a free to use software that cannot be modified and shared for profit;* ▪ *Shareware: free to use software that offers a limited functionality or based on a trial period;* ▪ *Subscription: is paid at regularly intervals, e.g. monthly or yearly;* ▪ *Perpetual: is paid once;* ▪ *Site: provides access to software on multiple or unlimited number of devices to be used in a single site, such as a University Campus;* ▪ *Open-Source: allows the user to use, modify and update the software as needed.*		

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
5.1e List different types of software licenses. *Limited to licenses in 5.1d.*	5.2e Distinguish between the different types of software licenses.	
	5.2f Recommend a type of license for a given scenario. *Limited to those listed in 5.1d.*	5.3f Justify the choice of license for a given scenario.
5.1g Define Software as a Service (SaaS). *Limited to:* ▪ *cloud-based service;* ▪ *accessing an application via an internet browser.*	5.2g Identify examples of SaaS in a given scenario. *Note: Background information will be provided with each example featuring in controlled assessment.* *Examples may include: online office apps, streaming content provider, cloud storage apps, etc…*	5.3g Justify the use of SaaS in a given scenario. *Note: Background information will be provided with each example featuring in controlled assessment.*

Subject Focus:	**Principles of Computing**
Learning Outcome 6: **(Paper II)**	**At the end of the programme, I can assess the suitability of different CPU specifications for different scenarios.**

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
6.1a Define the term CPU and/or 'CPU cycle'.	6.2a Describe the relationship between the CPU cycle and the system clock. *Note: This should not be limited to the theoretical basis but also applied to given scenarios.*	
6.1b Identify the basic components of the CPU. *Limited to Control Unit (CU) and Arithmetic Logic Unit (ALU).*	6.2b Describe the basic components of the CPU. *Limited to:* ▪ *Control Unit (CU);* ▪ *Arithmetic Logic Unit (ALU);* ▪ *Accumulator.*	6.3b Distinguish the function of the basic components of the CPU. *Limited to:* ▪ *Control Unit (CU);* ▪ *Arithmetic Logic Unit (ALU);* ▪ *Accumulator.*
6.1c List the basic components of the CPU. *Limited to Control Unit (CU) and Arithmetic Logic Unit (ALU).*		
6.1d Define a single core and/or a multi-core CPU.	6.2d Distinguish between a single core and a multi-core CPU.	6.3d Relate the system clock and/or the number of CPU cores to the overall system performance.
	6.2e Define machine instruction. *Limited to:* ▪ *the instruction that the CPU understands;* ▪ *opcode and operand.*	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
6.1f Identify the structure of a machine instruction. *Limited to:* ▪ *opcode and operand;* ▪ *questions in controlled assessment should be based on the Load/Store Instruction Set Architecture (RISC), to stay within the Von Neumann concept; i.e. read & execute one instruction at a time.* *Note: candidates are not expected to remember opcodes or addressing modes.* *For example:* *LOAD R1, num1* — *; copy the value in memory location called 'num1' to register R1.* *LOAD R2, 1101* — *; copy the value in memory location with address 1101 to register R2.* *SUB R1, 6* — *; subtract value 6 (00000110_2) from the value in register R1 and store result in R1.* *ADD R3, R1, R2* — *; add the values in registers R1 and R2 and store in register R3.* *STORE R3, 0x6B* — *; Store value in register R3 in memory location with address 6B {Hex value}.*	6.2f Distinguish between opcode and operand. *Limited to:* ▪ *opcode: instruction (action);* ▪ *operand: value, register or memory location.*	6.3f Describe the function of the instruction set. *Limited to:* ▪ *the entire set of instructions (opcodes) that the CPU understands;* ▪ *that different opcodes activates CPU circuitry differently to perform a specific task.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		6.3g Interpret a snippet of machine instructions. *Limited to:* ▪ *opcode and operand;* ▪ *questions in controlled assessment should be based on the Load/Store Instruction Set Architecture (RISC), to stay within the Von Neumann concept; i.e. read & execute one instruction at a time.* *Note: candidates are not expected to remember opcodes or addressing modes*
6.1h List the three main types of the system bus. *Limited to address, control and data bus.*	6.2h Describe the purpose of the system bus. *Limited to the connection between CPU and memory.*	
6.1i Classify the different types of buses. *Limited to 6.1f.*	6.2i Define the address and/or data and/or control bus. *Limited to:* ▪ *the purpose of each bus;* ▪ *uni/bi directionality of all buses;* ▪ *read/write line only for the control bus.*	
6.1j Define the Fetch and Execute Cycle. *Limited to: CPU getting an instruction from memory; and processes it, which may include storing the result back to memory.*		6.3j Outline the underlying concept of the Fetch and Execute CPU cycle (machine cycle). *Limited to:* ▪ *Control Unit fetches instruction from RAM,* ▪ *Control Unit decodes the instruction; i.e. gets required operands and prepares the instruction into commands that the CPU understands,*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		▪ *Control Unit, with the assistance of the ALU if needed, processes the instruction, and* ▪ *Control Unit stores result in a special-purpose register (Accumulator), or directly to a memory location according to the instruction.* *Note: Questions in controlled assessment may ask candidates to outline the concept of the fetch and execute cycle in terms of an instruction.* *Example 1:* *Fetch & Execute cycle for the instruction LOAD R1, num1; which copies the value in memory location called 'num1' to register R1.* ▪ *CU fetches the instruction from memory.* ▪ *CU decodes the instruction:* ○ *fetches the value in memory location num1, and* ○ *translates opcode LOAD into commands that the CPU understands.* ▪ *CU executes the commands to process the instruction.* ▪ *CU stores the result in accumulator R1.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		Example 2: *Fetch & Execute cycle for the first instruction ADD R3, R1, R2; which adds the value in register R1 by the value in register R2 and store in register R3.* ▪ *CU fetches the instruction from memory.* ▪ *CU decodes the instruction:* ○ *translates opcode ADD into commands that the CU understands.* ▪ *ALU executes the commands to process the instruction.* ▪ *CU stores the result in accumulator R3.* *Example 3:* *Fetch & Execute cycle for instruction STORE R3, 0x6B; which stores value in register R3 in memory location with address 6B {Hex value}* ▪ *CU fetches the instruction from memory.* ▪ *CU decodes the instruction:* ○ *translates opcode STORE into commands that the CU understands.* ▪ *CU executes the commands to process the instruction:* *sends the value in register R3 into memory location with address 01101011 ($6B_{16}$)*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		6.3k Explain the function of the system bus in terms of read and write cycles. *Limited to: LOAD and STORE instructions only.* *Examples: Instructions taken from example in 6.1e* ▪ *LOAD R1, num1;* ○ *Address Bus: CU sends address of the instruction that needs to process to memory (to fetch the instruction).* ○ *Data Bus: Instruction (LOAD R1, num1), from the memory location indicated by the address, is transferred to the CPU.* ○ *Control Bus: CPU indicates that a read operation from RAM is required.* ▪ *STORE R1, 0x6B;* ○ *Address Bus:* - *CPU sends address of the instruction that needs to process to memory (to fetch the instruction).* - *CPU sends the address of memory location 01101011 ($6B_{16}$) to memory (to store the processed instruction; i.e. the value of register R3)* ○ *Data Bus:* - *Instruction from the memory location indicated by the address; i.e. (STORE R3, 0x6B), is transferred to the CPU.*

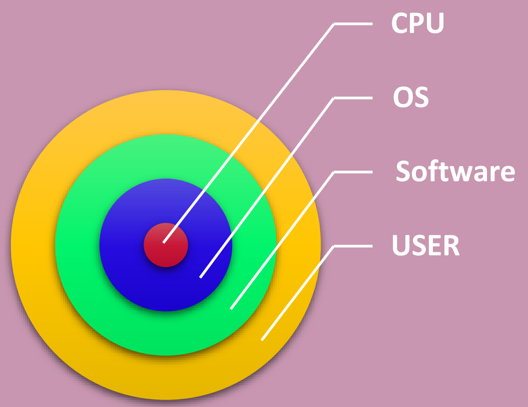
Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		- *Value stored in register R3, is transferred to memory location with address 01101011 ($6B_{16}$).* ○ *Control Bus:* - *CPU indicates that a read operation from RAM is required (to fetch the instruction).* - *CPU indicates that a write operation to RAM is required (to store value of register R3).*
6.1l Define address space. *Limited to the indication of the number of memory locations that the CPU can access.*	6.2l Describe the relation between the address bus and the address space.	
6.1m Define word length. *Limited to the indication of the number of bits of data can that the CPU handles (access from RAM and process) per CPU cycle.*	6.2m Describe the relation between the data bus and the word length.	6.3m Explain how the address space and/or word length impact system performance. *Examples:* 1. *an OS that requires a minimum of 128 MB of memory space will not run in a system that has a 16-bit address space.* *2^{16}= 65536 byte-addressable locations => total of 64 KB of RAM which is not sufficient to run the OS.* 2. *a 64-bit CPU performs better than a 32-bit CPU because it can handle (retrieve from memory and process) more bits of data per CPU cycle.* *Note: When controlled assessment involve the RAM, it must be indicated that it is byte-addressable memory.*

Subject Focus:	**Principles of Computing**
Learning Outcome 7: **(Paper II)**	**At the end of the programme, I can describe the role and the basic functions of an Operating System in different digital systems.**

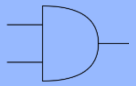
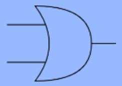
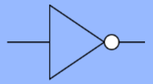
Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
7.1a Define the purpose of an Operating System (OS). *Limited to:* *1. managing of the device's resources, such as the CPU, memory, storage, and I/O devices;* *2. establish a user interface; and* *3. running of applications software.*		
7.1b Define a General-Purpose OS and/or Embedded OS.	7.2b Distinguish between General-Purpose OS and Embedded OS.	7.3b Recommend the type of OS (General-Purpose and/or Embedded OS) appropriate for a given scenario. *Note: In controlled assessment, descriptive scenarios should be provided; candidates should have enough details to answer related questions.* *Examples:* ▪ *OS for a photocopier,* ▪ *OS for a smart-phone,* ▪ *OS for a server running a central DB for a chain of markets,* ▪ *OS of an IP Camera (using IoT).*
		7.3c Justify the choice of OS type (General-Purpose and/or Embedded OS) for a given scenario. *Note: same as in 7.3b.* *Examples: same as in 7.3b.*
7.1d Define single and/or multitasking OS.	7.2d Distinguish between single and multitasking OS.	

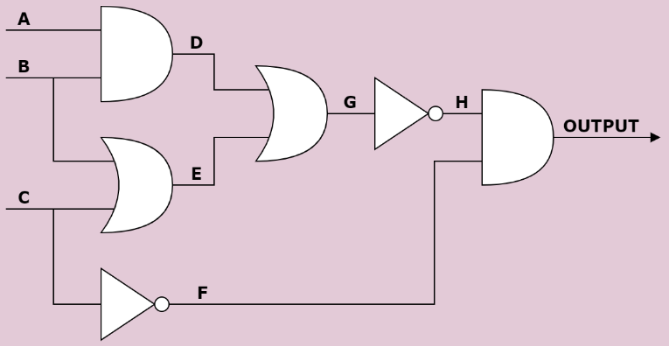
Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
7.1e List the basic functions of an OS. *Limited to: User Interface, File Management, Process Management, Memory Management, and Input & Output Management.*	7.2e Explain the need for the basic functions of the OS.	
7.1f Define User Interface (UI).		
7.1g List different types of UI. *Limited to:* Command Line Interface (*CLI),* Graphical-User Interface (*GUI) and* Natural User Interface (*NUI).*		
7.1h Identify different types of UI. *Limited to those listed in 7.1g.*		
7.1i Define the different types of UI. *Limited to those listed in 7.1g.*	7.2i Distinguish between the different types of UI. Limited to: ▪ *user needs (user experience vs limitations in functionality); and* ▪ *system resources needed.* *Examples: Data clerk requires a GUI for ease of use, network administrator requires a CLI for accessing certain functions, and a physically impaired individual requires a NUI with voice control.*	7.3i Recommend a UI for a given scenario.
		7.3j Justify the choice of UI for a given scenario. *Limited to those listed in 7.1g.*
7.1k Define a filing system.		

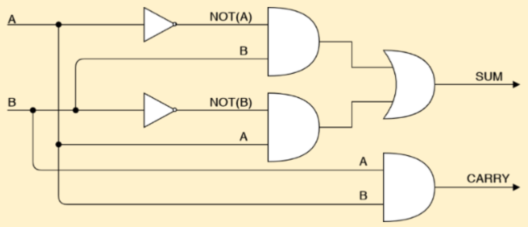
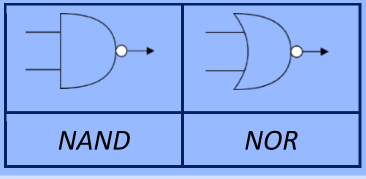
Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
7.1l List different filing systems. *Limited to: FAT32, NTFS, APFS, EXT4.*	7.2l Describe the use of filing systems.	
	7.2m Describe the use of virtual memory.	
	7.2n Describe the use of device drivers.	7.3n Justify the use of buffering to deal with devices with different speeds.
7.1o Define CPU process scheduling. *Limited to the allocation of time slices to different processes.*	7.2o Describe Round Robin and/or Priority scheduling. *Limited to: Non-preemptive priority scheduling.*	
	7.2p Distinguish between Round Robin scheduling and Priority scheduling.	
7.1q Define a system utility.		
7.1r List different system utilities. *Limited to:* *Anti-Virus / Anti-Malware, Backup/Restore, Disk Management (Clean Up and Defragmenter), File Compression, File Manager, Formatting, and Program Installer.*	7.2r Describe the use of different system utilities.	7.3r Recommend different system utilities according to a given scenario.
		7.3s Justify the use of different system utilities according to a given scenario. *Limited to those listed in 7.1r.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		7.3t Explain compatibility issues between the OS, CPU and application software. *Limited to compatibility issues related to CPU word-length, OS bit-size and application bit-size.* *Note: the below diagram aids in relating compatibility issues:*  - *To run an OS, it should be supported from the lower level (CPU).* - *To run an application, it should be support from all lower levels: OS and CPU.* *Examples:* ▪ *a 32-bit OS is supported by a 64-bit CPU but not vice-versa.* ▪ *a 64-bit application works only on 64-bit OS.*

Subject Focus:	**Machine Logic**
Learning Outcome 8: **(Paper II)**	**At the end of the programme, I can understand principles of machine logic in general and produce logic circuits, truth tables & Boolean expressions using the NOT, AND, and OR gates.**

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
8.1a Define a logic gate and/or a logic circuit and/or truth table. *Note: Definitions are limited to the below contexts:* ▪ *Logic Gates -> electronic components that performs logical operations.* ▪ *Logic Circuit -> several logic gates connected to provide an output according to certain criteria.* ▪ *Truth Table -> table showing all possible inputs and respective output.*		
8.1b Represent the AND, OR and NOT gates using standard symbols.  *AND*  *OR*  *NOT*		

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
8.1c Draw the truth table of the AND and/or OR and/or NOT Gate.	8.2c Draw the truth table for a given logic circuit and/or Boolean expression. *Limited to 3 inputs, 2-input gates, and using AND/OR/ NOT gates only.*	8.3c Derive the truth table to represent the solution for a given scenario. *Limited to 3 inputs, 3 intermediate steps, using 2-input gates, and using NOT/AND/OR gates only.*  *A, B & C => INPUTS* *D, E & F => 1st Intermediate Step* *G => 2nd Intermediate Step* *H => 3rd Intermediate Step*
8.1d Represent the AND and/or OR and/or NOT gates using Boolean expressions. ***Note:*** ▪ *AND represented with '**.**'* ▪ *OR represented with '**+**'* ▪ *NOT represented with '¯'* ▪ *Boolean expression can be represented using either symbols or keywords; e.g. X = A . B or X = A AND B*	8.2d Draw the logic circuit for a given truth table and/or Boolean expression. *Limited to those listed in 8.2c.*	8.3d Derive the logic circuit to represent the solution for a given scenario. *Limited to those listed in 8.3c.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	8.2e Express the Boolean expression from a given logic circuit and/or truth table. *Limited to those listed in 8.2c.*	8.3e Derive the Boolean expression to represent the solution for a given scenario. *Limited to those listed in 8.3c.*
	8.2f Complete a given logic circuit and/or truth table and/or Boolean expression. *Limited to those listed in 8.3c.*	8.3f Interpret a given logic circuit and/or truth table and/or Boolean expression. *Limited to those listed in 8.3c.*
8.1g Define half adder. *Note: Definition is limited to the concept of being an electronic circuit that adds two single-binary digits.*	8.2g Construct the truth table and/or logic circuit and/or Boolean expression of the half adder. *Note: Expected answers should only include a combination of AND, OR and NOT gates.* *Example:*  A NOT(A) B SUM B NOT(B) A A CARRY B	8.3g Solve a given scenario using concepts of the half adder. *Note: Description of given scenarios should be provided in examination questions.* *Example:* *The half adder can be represented by means of an XOR gate and an AND gate. The XOR gate receives the inputs and provides the sum value, while the AND gate receives the inputs and provide the carry value. Hence, if input A is 1 and input B is 0, the XOR gate gives a 1 output and the AND gate gives a 0 output. Construct the truth table of the XOR gate with inputs A and B.*
8.1h Represent the NAND and/or NOR gate using standard symbols and/or truth table.  *NAND* *NOR*	8.2h Describe the use of the NAND and/or NOR Gates as universal gates. *Note: 'Universal gates' refers to gates that can be used to produce all other gates.*	8.3h Explain the economical and manufacturing implications of the NAND and/or NOR gate as universal gates.
8.1i Identify logic gates from truth tables. *Limited to: AND, OR, NOT, NAND, NOR gates.*		

Subject Focus:	**Databases**
Learning Outcome 9: **(Paper II)**	**At the end of the programme, I can show an understanding of a relational database, its structure and use.**

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
9.1a Define a database.	9.2a Distinguish between manual record keeping and electronic databases.	
	9.2b Describe advantages and/or disadvantages of manual record keeping and electronic databases.	9.3b Explain the suitability of an electronic database in a given scenario.
9.1c Define a Database Management System (DBMS).	9.2c Distinguish between a DBMS and a database.	
9.1d Identify the key elements of a typical structure of a database. *Limited to tables, fields and records.*	9.2d Describe the key elements of a typical structure of a relational database. *Limited to tables, fields, field sizes, field types, primary keys, foreign keys and records.*	
9.1e Identify a key element of a typical structure of a relational database according to a given scenario. *Limited to those listed in 9.1d.*	9.2e Recommend a key element of a typical structure of a relational database according to a given scenario. *Limited to those listed in 9.2d.*	9.3e Justify the suitability of a key element of a typical structure of a relational database. *Examples:* ▪ *Quantity being a Number field type;* ▪ *Invoice Number being an Auto-Number field;* ▪ *Rental Date being Date/Time field type;* ▪ *Telephone Number being a Text field type;* ▪ *Foreign key being the exact field type of its related field.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	9.2f Interpret an ERD Diagram. *Limited to 3 tables and using crow's foot notation which includes: cardinality, optionality and indication of primary keys.* *Note: Refer to subject content section on page 104.*	9.3f Construct an ERD diagram for a given scenario.
9.1g Define data validation and/or integrity.	9.2g Explain the importance of data validation and/or integrity.	9.3g Recommend data validation rules and/or field sizes for specific fields. *Examples:* ▪ *Rental Date cannot be less than the current date.* ▪ *Telephone Number should only include 8 digits (numbers) only.*
		9.3h Justify the choice of a data validation rule and/or field size for specific fields.
9.1i Define a query.	9.2i Describe the use of Structured Query Language (SQL) in databases. *Limited to the creation of queries to extract data from a database using the SELECT and FROM and/or WHERE and/or ORDER BY only.*	
	9.2j Interpret a query. *Limited to a maximum of two conditions and one table.*	
	9.2k Modify a query. *Limited to a maximum of two conditions and one table.*	9.3k Construct a query to search for specific records by using SQL. *Limited to a maximum of two conditions and one table.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	9.2l Justify the role of a database as a back-end to IT applications. *Example: candidates should know that when using an online shop, the interaction happens through the front-end (app interface). The DB (back-end) is the structure where the data is stored.*	

Subject Focus:	**Fundamentals of Networking**
Learning Outcome 10: **(Paper II)**	**At the end of the programme, I can outline networking concepts of how devices communicate.**

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
10.1a Define a network. *Note: Communication should be considered between users and devices and devices with other devices. Devices should be considered in the broadest sense to include computers, mobiles, air conditioners, cars, washing machines, televisions and other devices.*	10.2a List advantages for networked devices as opposed to stand-alone devices.	
10.1b Identify Internet of Things (IoT) devices.	10.2b Describe the concept of the IoT.	10.3b Recommend examples of IoT devices.
10.1c Classify between wired and wireless connections. *Limited to:* ▪ *Wired: Ethernet, Fibre;* ▪ *Wireless: Wifi, Bluetooth, mobile data - 3G/4G/5G.*	10.2c Describe advantages and/or disadvantages between wired and wireless connections.	10.3c Justify a suitable connection (wired or wireless) for a given scenario. *Example:* ▪ *Ethernet is used into a smart TV, especially where an access point is placed far from the TV;* ▪ *WIFI is used at home where there is a WIFI connection;* ▪ *Mobile data is used in a public park where there is no access to a WIFI.*
	10.2d Recommend a suitable use according to the different networking connections of a device in relation to a given scenario. *Limited to those listed in 10.1c.*	10.3d Justify the choice of a suitable use according to the different networking connections of a device in relation to a given scenario.

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	10.2e Define the role of the client and/or server in network communication. *Limited to:* ▪ *the client is the device requesting the data, such as a web browser;* ▪ *the server is the device holding the data such as the files making up a website.* *Example: When browsing a website, the machine where the browser resides acts as a client to the machine where the server hosting the website resides.*	
10.1f Describe the need for a device to be uniquely identified in a network. *Limited to IP and MAC addresses.*	10.2f Distinguish between IP and MAC address.	
	10.2g Distinguish between IPv4 and IPv6. *Limited to IPv4 being 32-bit address and IPv6 being a 128-bit address.*	10.3g Justify the need for IPv6. *Example: More devices communicating together including the IoT.*
10.1h Define the need for a network protocol as a method of communication between different devices.	10.2h Justify the need of host names and/or translation service between host names and IPs. *Example: www.google.com translates to 216.58.205.196, hence easier for users to remember. Also, host name remains the same even when there is a server change (change in IP).*	
10.1i Define HTTP and/or FTP protocol.	10.2i Distinguish between FTP and HTTP.	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
10.1j Define the term firewall.	10.2j Explain the need for security in networking. *Limited to:* ▪ *The use of digital certificates and encryption (HTTPs);* ▪ *Firewall.* *Example: Accessing a private network through a VPN.*	10.3j Justify the need for security in a given scenario. *Example: HTTPs is required for user authentication and bank details but not to display a page in a blog.*
10.1k List different hardware components which are needed by devices to communicate. *Limited to Network Interface Card (NIC), Router, Access Points and Switch.*	10.2k Recommend suitable network components required for a given scenario.	10.3k Justify the use of suitable network components required for a given scenario. *Example: the components needed to build a home wireless network.*
10.1l List different types of networks. *Limited to:* ▪ *Wired and wireless connections;* ▪ *Local Area Network (LAN);* ▪ *Wide Area Network (WAN);* ▪ *Personal Area Network (PAN) connections.*	10.2l Distinguish between the different types of networks.	
10.1m Define different types of networks. *Limited to those listed in 10.1l.*	10.2m Recommend the use of a type of network for a given scenario.	10.3m Justify the use of a type of network for a given scenario.

Subject Focus:	**Problem Solving, Algorithms and Physical Computing**
Learning Outcome 11: (Paper I and Paper II)	**At the end of the programme, I can produce algorithms, by applying problem solving concepts, and develop simple dedicated systems using boards powered by microcontrollers or System on Chip (SoC) technology.** *Note:* ▪ *Private candidates are to follow Appendix 1 for further details;* ▪ *It is not expected that candidates write programs in Paper II.*

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
11.1a Define the term algorithm.		
11.1b Define pseudocode and/or flowcharts.	11.2b Distinguish between pseudocode and flowcharts.	
11.1c Identify symbols used in flowcharts. *Limited to: start/end, input/output, process, decision, connector symbols and flow of sequence.*		
11.1d Define symbols used in flowcharts. *Limited to symbols listed in 11.1c.*		
11.1e Define the basic programming constructs. *Limited to sequence, selection and iteration.*	11.2e Complete the missing parts of a given algorithm. *Limited to: input/output, process, selection and iteration instructions.* *Note:* ▪ *Algorithms can be represented in pseudocode or flowchart;* ▪ *Candidates are not expected to follow any pseudocode standard.*	11.3e Interpret a given algorithm. *Limited to those listed in 11.2e.* *Notes as those mentioned in 11.2e.*

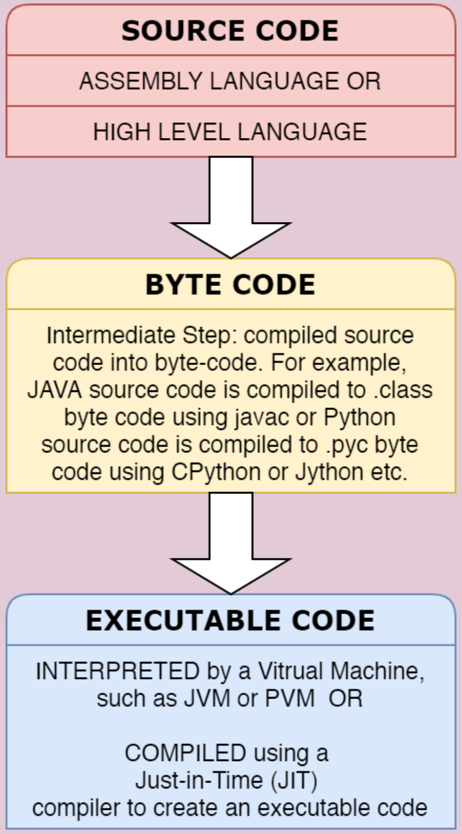
Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		11.3f Construct an algorithm for a given scenario. *Limited to those listed in 11.2e.*
11.1g Define syntax and/or logical and/or runtime errors.	11.2g Distinguish between syntax, logical and runtime errors.	
	11.2h Explain the effect of an error on program compilation and/or runtime. *Limited to:* ▪ *syntax: does not compile;* ▪ *logical: gives undesired result;* ▪ *runtime: stops executing.*	
11.1i Develop a program for a dedicated system that includes output and/or input constructs using digital components only. *Limited to:* ▪ *the use of single-board systems, bread board and text-based programming language, such as C++, Python, etc.* ▪ *Digital Components: LEDs, buzzer & push switch.* *Note: Candidates should be able to declare and use integer and Boolean variable types.*	11.2i Develop a program for a dedicated system that includes analogue output using digital components only. *Note that this criterion includes knowledge of the Pulse Width Modulation (PWM) to simulate analogue output on digital components.* *Example: Fading LED or RBG LED.*	11.3i Develop a program for a dedicated system that includes output and/or input constructs using both digital and/or analogue components. *Limited to at least one analogue component that does not require the use of libraries, such as: potentiometer, photo resistor, sound sensor, accelerometer, analogue joystick module, etc.* *Note that this criterion includes knowledge of the sampling rates for digital and analogue pins and the conversion from one form to the other. For example: Arduino's digital pins are 8-bits while analogue pins are 10-bits. The use of the map function (in C++) may aid in converting the sampled data entered into the analogue pins to data sent from digital pins.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	11.2j Develop a program for a dedicated system that includes arithmetic and assignment operators. *Limited to the following:* *Arithmetic operators:* **=** *Equals* **+** *Addition* **-** *Subtraction* ***** *Multiplication* **/** *Division* *Assignment operators:* **+=** *x=x+3* **-=** *x=x-3* ***=** *x=x*3* **/=** *x=x/3* **++** *x=x+1* **--** *x=x-1*	
	11.2k Develop a program for a dedicated system that includes selection constructs. *Limited to: if, if-else and else if constructs.*	11.3k Develop a program for a dedicated system that includes nested selection constructs.
	11.2l Develop a program for a dedicated system that includes iteration constructs. *Limited to: for and/or while loops.* *Note: If Arduino is used, the default loop() method does not satisfy this criteria.*	11.3l Develop a program for a dedicated system that includes nested iteration constructs.

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
		11.3m Develop a program for a dedicated system that includes the use of libraries. *Limited to at least one library for the following components: Servo, Distance Sensor, Temperature Sensor, LCD Display (preferably with i2C module).* *Note: It is suggested that the following libraries are used in the case that Arduino microcontroller is used: Servo built-in library, HCSR04, TMP36 and LiquidCrystal_I2C.*
		11.3n Trace values of analogue sensors in real time. *Limited to transferring data from dedicated system to device.* *Note:* ▪ *If Arduino is used, the serial-monitor and the Serial function are ideally used;* ▪ *If other systems are used, a third-party tool may be used, such as the MicroPython, Python Mu editor, ppLOGGER, etc.*

Subject Focus:	**Programming Languages and Fundamentals to Program Development**
Learning Outcome 12: **(Paper II)**	**At the end of the programme, I can distinguish between low and high-level languages.**

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
12.1a Define Low-Level Language (LLL) and/or High-Level Language (HLL). *Note: definitions limited to the following concepts:* ▪ *LLL:* ○ *contain basic instructions understood by the computer, thus more difficult to program with;* ○ *dependent on the processor.* ▪ *HLL:* ○ *more level of abstraction from the processor, thus more user friendly;* ○ ***independent to the processor, thus offering software portability.***	12.2a Distinguish between LLL and/or HLL. *Characteristics limited to:* ▪ *LLL:* ○ *Machine Language: Machine-dependent language, language structure follows instruction set architecture (i.e. opcode and operand), and there's no need of translation.* ○ *Assembly Language: same as Machine language but uses symbolic notations (MNEMONICS) and require translation to Machine Language.* ▪ *HLL:* ○ *Third Generation Languages (3GL): Machine Independent, English-like statements, require translation to Machine Language. Examples include C, JAVA, Python, etc.* ○ *Fourth Generation Language (4GL): Same as 3GL but easier to use (using English-like words and phrases, and/or the use of icons, graphical interfaces and symbolical representations). Designed to reduce the overall time for software development but do not offer the flexibility of 3GL. Examples include: Structured Query Language (SQL), Prolog, VPLs, GUI creators, etc.*	12.3a Justify the use of a specific type of programming language for a given scenario. *Example: A smart air-conditioner is programmed in assembly language but its mobile app in 3GL.*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
12.1b Classify languages in LLL and/or HLL. *Limited to Assembly, JAVA, Python, C and SQL.*		
12.1c Define source code and/or executable code. *Limited to:* ▪ *source code: the code developed by the programmer;* ▪ *executable code: the code which is understood by the device running it.*	12.2c Distinguish between byte code and executable code *Limited to: byte code offers platform compatibility and executable code runs faster.*	12.3c Explain the need for code translation, including the relation between the source code, byte code and executable code. *Limited to:*  **SOURCE CODE** ASSEMBLY LANGUAGE OR HIGH LEVEL LANGUAGE **BYTE CODE** Intermediate Step: compiled source code into byte-code. For example, JAVA source code is compiled to .class byte code using javac or Python source code is compiled to .pyc byte code using CPython or Jython etc. **EXECUTABLE CODE** INTERPRETED by a Vitrual Machine, such as JVM or PVM OR COMPILED using a Just-in-Time (JIT) compiler to create an executable code

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
12.1d List the different types of language translators. *Limited to assembler, interpreter and compiler.*	12.2d Distinguish between the different types of language translators. *Limited to:* ▪ *translation of instructions from: Assembly to Machine Language and High level to Machine language;* ▪ *interpreter: line by line translator at runtime;* ▪ *compiler: provides executable code and error log.*	12.3d Recommend a language translator for a given scenario.
		12.3e Justify the choice of language translator for a given scenario.

Subject Focus:	**Programming Languages and Fundamentals to Program Development**
Learning Outcome 13: (Paper I and Paper II)	**At the end of the programme, I can develop programs using Python programming language that includes textual and graphical interfaces.** *Note that it is not expected that candidates write programs in Paper II.*

Assessment Criteria (LEVEL 1)	**Assessment Criteria (LEVEL 2)**	**Assessment Criteria (LEVEL 3)**
13.1a Interpret a program snippet that includes output statements. *Limited to print() function without output formatting.*		13.3a Interpret a program snippet that includes output and /or formatted output statements. *Limited to:* ▪ *new line & no new line in print function;* ▪ *the use of the F-Strings or String Concatenation (String+Var).*
13.1b Develop a program using output statements according to a given scenario. *Limited to those listed in 13.1a.*		13.3b Develop a program snippet that includes output and/or formatted output statements according to a given scenario. *Limited to those listed in 13.3a.*
13.1c Define a variable. *Limited to: in terms of reserving a memory location to store a value.*		
13.1d Identify between different types of variables for a given scenario. *Limited to: integer, float, String and Boolean.*		

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
13.1e Interpret a program snippet that includes the use of variables and/or input statements. *Limited to:* ▪ *input() function;* ▪ *variables as listed in 13.1d.* *Note: program-snippets may include variable initialization, such as:* `num1=5` *or* `name="Anne"`.	13.2e Interpret a program snippet that includes the use of variables and/or type conversion functions and/or input statements. *Limited to int(), float(), and str() type conversions.*	
13.1f Develop a program that includes the use of variables and/or input statements. *Limited to those listed in 13.1e.*	13.2f Develop a program using variables and/or type conversion functions and/or input statements. *Limited to those listed in 13.2e.*	
13.1g Identify arithmetic operations in a given scenario. *Limited to the following arithmetic operators:* = *equals*; + *addition*; - *subtraction*; * *multiplication*; / *division*; // *floor division*; % *modulus*; ** *power of*	13.2g Interpret a program snippet that includes arithmetic operations.	13.3g Interpret a program snippet that includes assignment operators. *Limited to:* += *x=x+3*; -= *x=x-3*; *= *x=x*3*; /= *x=x/3*; //= *x=x//3*; %= *x=x%3*
	13.2h Develop a program using arithmetic operations. *Limited to those listed in 13.1g.*	13.3h Develop a program using assignment operators. *Limited to those listed in 13.3g.*

Arithmetic operators (13.1g):

=	*equals*		/	*division*
+	*addition*		//	*floor division*
-	*subtraction*		%	*modulus*
*	*multiplication*		**	*power of*

Assignment operators (13.3g):

+=	*x=x+3*		/=	*x=x/3*
-=	*x=x-3*		//=	*x=x//3*
*=	*x=x*3*		%=	*x=x%3*

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	13.2i Interpret a program snippet that includes decision statements. *Limited to:* ▪ *Decision statements: if, elif, else* ▪ *Conditional operators: ==, !=, >, <, >=, <=* ▪ *Logical Operators: and, or, not* ▪ *Membership operators: in, not in*	13.3i Interpret a program snippet that includes nested decision statements.
	13.2j Develop a program using decision statements. *Limited to those listed on 13.2i.*	13.3j Develop a program using nested decision statements.
	13.2k Interpret a program snippet that includes iteration statements. *Limited to: for-in loop, while loop, and while-else loop.* *Note: the for-in loop may include the range() function.*	13.3k Interpret a program snippet that includes nested iteration statements. *Note: may include 'break' and 'continue' statements.*
	13.2l Develop a program using iteration statements. *Limited to those in 13.2k.*	13.3l Develop a program using nested iteration statements.
13.1m Define the term function and/or module. *Note:* ▪ *Function refers to the isolation of a part of the program's functionality and make it reusable;* ▪ *Module refers to the collection of functions in one file which can be used in various projects.*	13.2m Distinguish between function and module.	
	13.2n Distinguish between built-in function and user-defined function.	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	13.2o Interpret a program snippet that includes String functions. *Limited to: Get character from string (String[position]), Substring (String[2:5]), strip(), len(), lower(), upper(), replace(), find(), count().*	13.3o Interpret a program snippet that includes user-defined functions. *Limited to return with only 1 value and positional arguments only*
	13.2p Develop a program using String functions. *Limited to those in 13.2o.*	13.3p Develop a program using user-defined functions. *Note: user-defined functions may include value return and parameters.*
	13.2q Interpret a program snippet that includes tuples and/or lists as data structures. *Limited to the use of the following functions: append(), clear(), count(), insert(), len(), pop(), remove(), reverse(), and sort().*	13.3q Interpret a program snippet that includes dictionaries as data structure. *Limited to the use of the following functions: clear(), items(), get(), and pop().*
	13.2r Develop a program using tuples and/or lists as data structures. *Limited to those in 13.2q.*	13.3r Develop a program using dictionaries as data structures. *Limited to those in 13.3q.*
	13.2s Interpret the statement import and/or from-import and/or from-import-as. *Examples:* ▪ *import random* ▪ *from random import randint* ▪ *from random import ** ▪ *from random import randrange as num_generator*	

Assessment Criteria (LEVEL 1)	Assessment Criteria (LEVEL 2)	Assessment Criteria (LEVEL 3)
	13.2t Interpret a program snippet that includes the Random module. *Limited to randint(start, stop) function.*	13.3t Interpret a program snippet that includes the Random module to access and modify items in data structures. *Limited to: choice(), shuffle() and sample() from a List or Tuple only.*
	13.2u Develop a program using the Random module. *Limited to those in 13.2t.*	13.3u Develop a program using the Random module to access and/or modify items in data structures. *Limited to those in 13.3t.*
	13.2v Develop, with assistance, a program that includes a graphical interface using any readymade module. *Example: Turtle module or Tkinter module or PyQt module, etc.* *Note: Candidates are given assistance from teachers to develop the requested program, such as completing a given source code.*	13.3v Develop a program that includes a graphical interface using any readymade module.
	13.2w Identify syntax and/or logical and/or runtime errors in a given algorithm.	13.3w Solve syntax and/or logical and/or runtime errors in a given algorithm.
13.1x List good programming practices. *Limited to inline comments, code indentation and meaningful variable names.* *This criterion also applies to LO11.*	13.2x Describe the importance of good programming practices. *This criterion also applies to LO11.*	
	13.2y Implement good programming practices. *Limited to those listed in 11.1r.* *This criterion also applies to LO11.*	

Scheme of Assessment

General Notes

- Some assessment criteria include further information in italics as follows:
    - “Limited to” implies that only those examples listed will be examined.
    - “Example/s” means that apart from the example/s listed, other related instances may be examined.
    - “Note” means that candidates need to be aware of other details related to the respective assessment criteria

  Such information is stated in only one criterion per row. If not stated otherwise, it also affects the adjacent criteria within the same row.
- Questions in both controlled and coursework assessments will be set in English and must be answered in English.
- Electronic calculators must **not** be used in any part of the controlled assessment.
- Flowchart templates may be used in any part of the controlled assessment.
- The following handouts will be provided in level 1-2-3 Controlled Assessment I paper:
    - “Arduino: Sample Circuits and Code”; found in Appendix 2.
    - “Python Code Constructs”; found in Appendix 3.

School Candidates

The assessment consists of Paper I and Paper 2. Paper I consist of unmoderated school-based assessment (SBA) that is to be set and assessed by the school. Paper II consists of a controlled assessment that will take place at the end of the three-year programme.

School-based assessment (SBA): is any type of assessment of a candidate made by the school relevant to the respective SEC syllabus contributing to the final level awarded in the subject.

Controlled assessment: is comprised of a two-hour written exam set at the end of the programme and differentiated between two tiers:

a. Levels 1 and 2;
b. Levels 2 and 3.

Candidates are to satisfy the examiner in Paper I and Paper II to obtain a level higher than 1.

Paper I - School Based Assessment (30% of the total mark).

The school-based assessment shall be marked out of 100 each year (9, 10 and 11). The assessment for each year will contribute to 10% of the overall mark and will be reported to MATSEC by the school in Year 11. Therefore, each year will equally contribute to the final mark of the school-based assessment. The school-based assessment shall reflect the MATSEC syllabus covered in Year 9, Year 10 and Year 11.

School-based assessment can be pegged at either of two categories:

- SBA at categories 1-2 must identify assessment criteria from these two levels. It is suggested that ACs are weighted at a ratio of 40% at Level 1 and 60% at Level 2.
- SBA at categories 1-2-3 must identify assessment criteria from each of Levels 1, 2, and 3. It is suggested that ACs are weighted at a ratio of 30% at each of Levels 1 and 2, and 40% at Level 3.

The mark for SBA at level categories 1-2 presented for a qualification at level categories 2-3 will be calculated to 60% of the original mark. The mark stands in all other cases.

Paper II - Controlled Assessment (70% of the total mark).

Written Examination (100 marks; 2 hours)

Learning outcomes with assessment criteria in the psychomotor domain can be assessed by asking questions in pen-and-paper format seeking understanding of the activity.

Controlled Assessment will:

- consist of questions of either level 1-2 or level 2-3.
- cover most learning outcomes, including all learning outcomes which are not indicated to be covered through coursework.
- be marked out of 100 and all questions in each section are compulsory. Answers are to be written on the examination paper provided.
- consist of two obligatory sections, as follows:
  - Section A: up to five questions totalling 40 marks (possibly scenario-based).
  - Section B: three scenario-based questions totalling 60 marks.

Private Candidates

Private candidates will not be expected to carry out any school-based assessment as school candidates. Instead, private candidates need to sit for another Controlled paper as an alternative to the school-based assessment. Private candidates will be assessed through the means of **TWO** Controlled papers, one of which is common with school candidates.

Paper I – Controlled Assessment - Private Candidates Only (30% of the total mark).

Written Examination (100 marks; 2 hours)

Paper I for private candidates shall be a controlled assessment assessing levels 1, 2 and 3 as described in the respective syllabus and set and marked by MATSEC. It shall mainly focus on the learning outcomes marked in the respective syllabi as suggested for school-based assessment.

Learning outcomes with assessment criteria in the psychomotor domain can be assessed by asking questions in pen-and-paper format seeking understanding of the activity.

Controlled Assessment will:

- assess all learning outcomes which were indicated as part of school candidates' SBA and some other outcomes;
- have two sections as follows:
    - Section A:
        - One scenario-based question on learning outcome 11, ranging 40 - 60 marks.
        - Answers in this section are to be written on the examination paper provided.
    - Section B:
        - A scenario-based question that requires candidates to develop a program with a text-based interface using Python programming language ranging 40 - 60 marks.
        - Answers in this section are to be developed using Thonny IDE.

Controlled Assessment I will be marked out of 100 and all questions in each section are compulsory.

Paper II - Controlled Assessment (70% of the total mark).

Paper II is common with school candidates.

Appendices

Appendix 1: Private Candidates

For controlled assessment I, private candidates are to follow the below points:

Question related to learning outcome 11:

- Candidates will not be requested to write or develop any code. Candidates can only be asked to interpret sub-questions related to the source code for a given scenario.
- Arduino UNO board is to be used.
- The official Arduino language (set of C/C++ libraries) is to be used.
- Digital components to be used are: LED, RGB LED, Buzzer and Micro Switch.
- Analogue components to be used are: potentiometer and photocell sensor (LDR).
- External Library to be used is the Servo Arduino library.

Questions related to learning outcome 13:

- Candidates are to use Thonny IDE to develop code.

MATSEC reserves the right to make changes to the microcontroller-powered board and the programming language used if they are not available at the time of registration. An official communication will be published on the MATSEC website.

Appendix 2: Arduino: Sample Circuits and Code

Sample Code:

LED

- Switch ON an LED connected to pin 7: `digitalWrite(7, HIGH);`
- Switch OFF LED connected to pin 7: `digitalWrite(7, LOW);`

RGB LED

- The below example generates a RED colour on the RBG LED which is connected on pin 11 (RED), pin 10 (green) and pin 9 (blue):

```
analogWrite(11, 255);
analogWrite(10, 0);
analogWrite(9, 0);
```

Buzzer

- Buzzer makes a sound: `tone(Pin, Frequency, Duration);`
- `tone(3, 1000);` - Code that sends a 1000 sound frequency to the buzzer which is connected to pin 3.
- `tone(5, 1400, 2000);` - Code that sends a 1400 sound frequency for 2 seconds to the buzzer which is connected to pin 5.
- Buzzer connected to pin 8 is set to silent: `noTone(8);`

Micro Push Switch

The below example reads signal from the switch connected to pin 5 and stores the value in variable named buttonState: `int buttonState = digitalRead(5);`

Analogue Component: Potentiometer and Photocell Sensor

The below example reads analogue signals from an analogue input component connected to pin A1 and stores the value in a variable named inputValue: `int inputValue = analogRead(A1);`

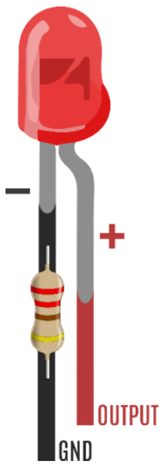
Servo

This code turns the servo to 90^{o}:

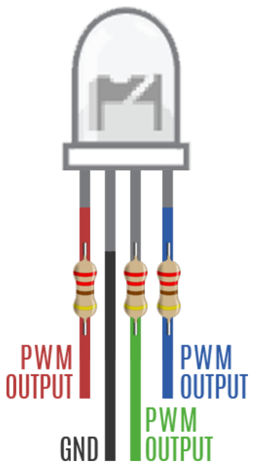
```
#include <Servo.h>
Servo myServo;
void setup() {
   myServo.attach(9);
   myServo.write(90);
}
```

Sample Circuits

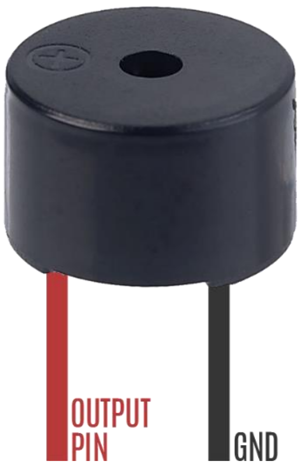
LED



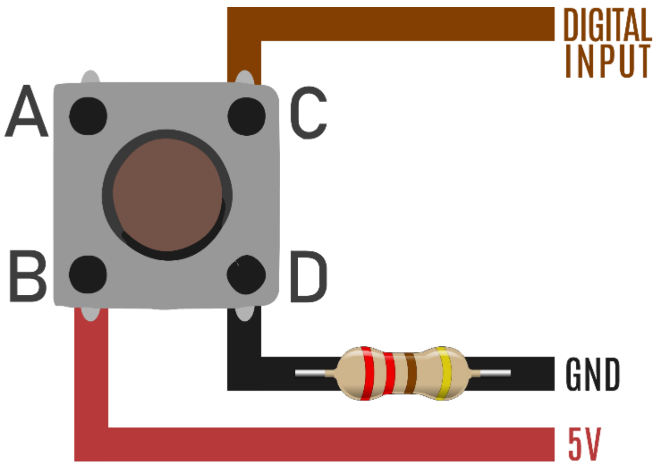
RGB LED



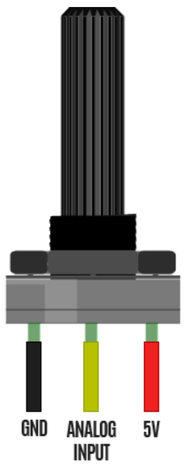
BUZZER



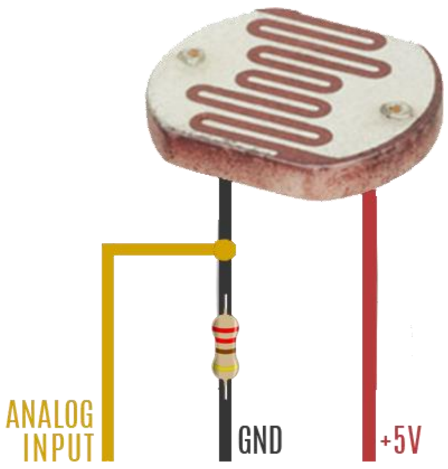
MICRO PUSH SWITCH



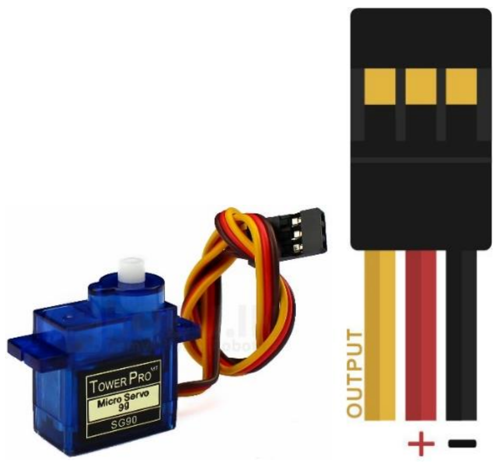
POTENTIOMETER



PHOTOCELL SENSOR



SERVO



Appendix 3: Python Code Constructs

- Comments:

Example1: One-line comment:

```
# This is a comment
```

Example 2: Block Comment:

```
"""
This is a
multiline comment
"""
```

- Output Statements: `print("This is a message")`

Example1: Skip Line

```
print("This is a message")
```

Example 2: Without skipping line

```
print("This is a message", end="")
```

- Formatted Output statement:

Example:

```
mystring = "hello"
mystring2 = "world"
myfloat = 10.765

print(f"Float with 2 decimal places: {myfloat:.2f}")
print(f"String: {mystring} dear {mystring2}")
```

- Input Statement: `user_name = input("Enter Name: ")`

Note: Input statement always return a String value

- Data Type conversions: int() or float() or str()

Example 1: Output message concatenation

```
one = str(1)
two = str(2)
hello = "hello"
print(one + two + hello)
```

Example 2: Inputting integer values

```
num1 = int(input("Enter Num1: "))
num2 = int(input("Enter Num2: "))
total = num1 + num2
```

- Arithmetic operators:

=	+	-	*	/	//	%	**
Equals	Addition	Subtraction	Multiplication	Division	Floor Division	Modulus	Power of

- Assignment operators:

+=	-=	*=	/=	//=	%=
x = x+3	x = x-3	x = x*3	x = x/3	x = x//3	x = x%3

- Conditional operators: ==, !=, >, <, >=, <=

==	!=	>	<	>=	<=
Equals	Not Equal	Greater than	Smaller than	Greater or equal to	Smaller or equal to

- Logical Operators: and, or, not

and	or	not

- Membership operators: in, not in

in	not in

- Decision Statements: if /elif / else Constructs

Example 1: if - else

```
x = 2
if x == 2:
    print("x equals two!")
else:
    print("x does not equal to two.")
```

Example 2: if - elif - else

```
x = 2
if x == 1:
    print("x equals one")
elif x == 2:
    print("x equals two")
else:
    print("x is not valid!")
```

Example 3: Multiple Conditions

```
name = "John"
age = 23
if name == "John" and age == 23:
    print("John, you are 23 years old.")
else
    print("You are not the 23 year old John!")
```

Example 4: Using membership operators

```
message = "Testing message"
if "test" in message:
    print("Keyword found")
else:
    print("Keyword not found")
```

- Loops: for / while

Example 1: For Loop – display numbers from 1 to 5

```
for x in range(1, 6):
    print(x)
```

Example 2: While Loop – display numbers from 1 to 5

```
count = 1
while count <= 5:
    print(count)
    count += 1
```

Example 3: While Loop – using 'break' statement

```
count = 1
while True: #infinite loop
    print(count)
    count += 1
    if count >= 5:
        break
```

Example 4: While Loop – using 'continue' statement

```
#Prints only odd numbers
for x in range(10):
    if x % 2 == 0: #Check if x is even
        continue
    print(x)
```

- String functions:

Example:

```
mystring1 = “This is a”

mystring2 = “message”


print(mystring1[8]) #prints “a” – 9th position of mystring1

print(mystring1[0:3]) #prints “Thi” – 1st till 3rd position of mystring1(index
0 till index 2)

print(len(mystring2)) #prints 7 – number of characters in mystring2

print(mystring2.index("g")) #prints 5 – ‘g’ found in 6th position of mystring2

print(mystring1.count("s")) #prints 2 - there are 2 's' in mystring1

print(mystring2.upper()) #prints mystring2 in uppercase

print(mystring1.lower()) #prints mystring1 in lowercase

print(mystring2.replace("message", "paragraph")) #replaces the word message
with paragraph

mylist = mystring1.split(" ") #crates a list with words from mystring1 split
at empty spaces
```

- User-Defined Functions:

Example 1: with no parameters and no value return

```
a=2
b=3

def print_total():
    print(a+b)

print_total()
```

Example 2: with value return but no parameters

```
a=2
b=3

def calculate_total():
    return a+b

print(calculate_total())
```

Example 3: with value return and parameters

```
def calculate_total(num1, num2):
    return num1 + num2

print(calculate_total(2, 3))
```

- Data Structures:

Example 1: Print items in a List

```
mylist = [12, Jack, 26]

for i in list:
    print(i)
```

Example 2: Add item in a List

```
mylist = [12, Jack, Lemon]
mylist.append(“Apple”)
```

Example 3: Delete item from List

```
mylist = [12, Jack, Lemon]
#removes item Jack
mylist.remove(“Jack”)
#removes item at index 1
mylist.pop(1)
```

Example 4: Displaying keys and values in a dictionary

```
phonebook = {"John": 938477566, "Jack": 938377264, "Jill": 947662781}
for name, number in phonebook.items():
    print(f "Phone number of {name} is {number}")
```

Example 5: Search a keyword in a dictionary and get its value

```
phonebook = {"Johnny": 938477566, "Jack": 938377264,
             "Jill": 947662781, "John": 98745632}
if phonebook.get("John"):
        print(phonebook.get("John"))
```

Example 6: Delete an item from a dictionary

```
phonebook = {"Johnny": 938477566, "Jack": 938377264,
             "Jill": 947662781, "John": 98745632}
if "John" in phonebook:
    phonebook.pop("John")
```

Example 7: Add items to a dictionary

```
phonebook = {"Johnny": 938477566, "Jack": 938377264,
             "Jill": 947662781, "John": 98745632}
phonebook["Mary"] = 92524217
```

- Random Module

Example 1: Display a random number between 1 and 10

```
import random
my_num = random.randint(1, 10)
print (my_num)
```

Example 2: Display a random item from List

```
import random
my_list = ["Lemon", "Apple", "Orange", "Banana"]
print (random.choice(my_list))
```

Example 3: Display a reshuffled List

```
import random
my_list = ["Lemon", "Apple", "Orange", "Banana"]
random.shuffle(my_list)
print (f"Reshuffled list: {my_list}")
```

Example 4: Generate 5 random non-duplicate numbers from 1 to 100 and store in list

```
import random
nums_generated = random.sample(range(1, 101), 5)
```

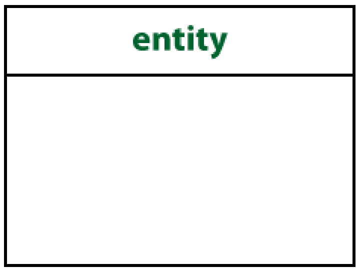
Subject Content

Crow's Foot Notation

Tables

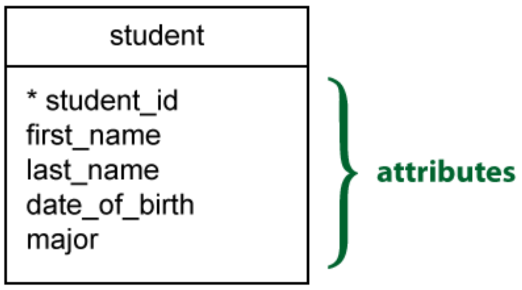
A table, or the so-called entity, is a representation of a class of object. It can be a person, place, thing, etc.

A table is represented by a rectangle, with its name on the top. The name is singular (entity) rather than plural (entities).



Fields

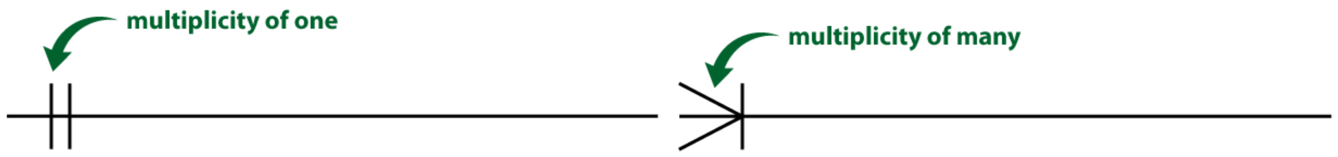
Fields, or the so-called attributes, are properties that describes a table. The primary key is marked with an asterisk.



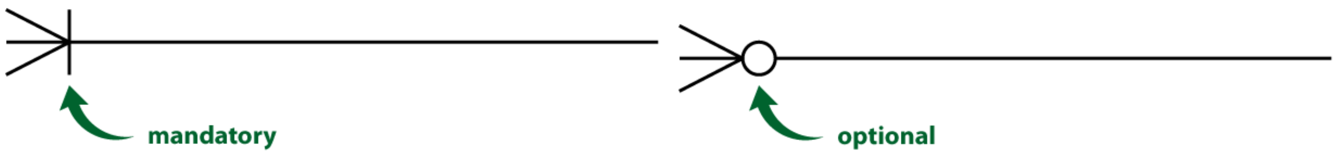
Relationships

Relationships have two indicators:

- The first one (called cardinality) refers to the maximum number of times that an instance of one table can be associated with instances in the related table. It can be one or many.

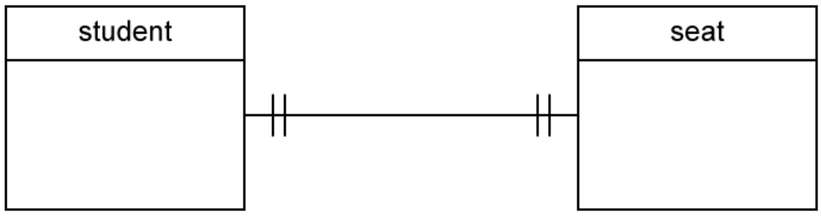


- The second (called optionality) describes the minimum number of times one instance can be related to others. It can be zero or one, and accordingly describes the relationship as optional or mandatory.

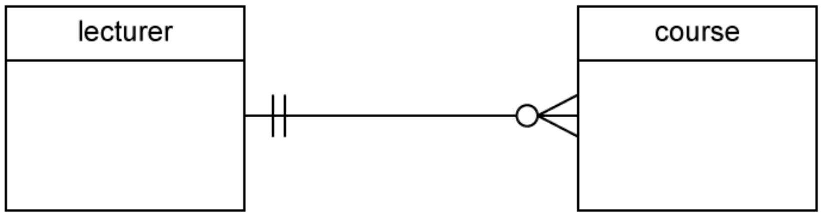


Examples of relationships:

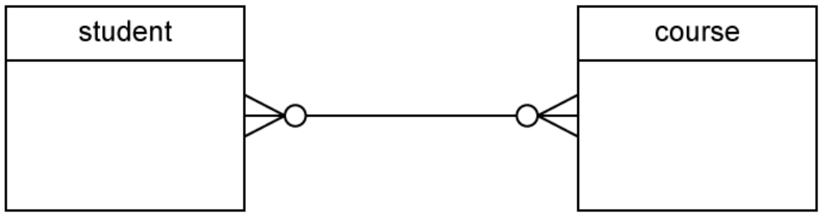
- One-to-one



- One-to-many



- Many-to-many



References:

Dybka, P. (2016, March 31). Design Fundamentals: Crow's Foot Notation. Retrieved from https://www.vertabelo.com/blog/crow-s-foot-notation/