

**IM SYLLABUS (2011)**

**COMPUTING**

**IM 07**

***SYLLABUS***

<b>Computing IM 07 Syllabus</b>	(Available in September) 1 Paper (3 hours) + Coursework
-------------------------------------	--

## 1. INTRODUCTION

This MATSEC Intermediate level Computing Syllabus has been prepared and compiled in line with previous syllabi, latest Computing related developments and space for future syllabi to add and enhance the contents. It is intended as a natural progression from SEC level Computer Studies and covers a reasonable and coherent portion of the MATSEC Advanced level Computing syllabus. It is aimed at eliciting roughly one third of the effort required in the MATSEC Advanced level Computing syllabus. It should be stressed that this is not a Computer Appreciation or a Computer Applications course. Furthermore, in line with the objectives clearly set in the previous version of this syllabus, the programming language that is to be used throughout this syllabus is to be “Java”. This language will displace the use of the “Pascal” programming language in every aspect of this syllabus wherever the use of a programming language is required, including the implementation of the programming assignment.

It is also stressed that all theoretical treatment of topics should be adequately accompanied by practical (real-world) examples when and wherever applicable.

Candidates are expected to have followed the Computer Studies stream at secondary level and have progressed on to the SEC level Computer Studies thereby acquiring sound background knowledge of the history of Computing, current and future trends, as well as the sector-specific and social applications of Information Technology.

This syllabus is ideal for those students who wish to deepen their understanding of Computing, not down to the MATSEC Advanced Computing level but enough to give a basic understanding of both the theory and techniques used in computing, together with the ability to use a programming language and basic application packages in practical situations.

This document is organized as follows. The next section briefly underlines the contents of the examination itself while syllabus details are described in detail in Section 3. Section 4 lists recommended texts and reference books that Computing educators can make use of to assist their students. Finally, detailed information regarding the assignments is expanded in Section 5.

## 2. EXAMINATION

The examination shall consist of two parts, namely, one written paper of three hours duration and one school-based programming assignment moderated by the MATSEC Markers’ Panel.

The written paper (80%) will be divided in two sections, namely:

- a) Section A (60 marks) will consist of ten compulsory short questions requiring to the point short answers.
- b) Section B (20 marks) will consist of two more in-depth questions of which the candidates will be required to answer one.

The Set Programming Assignment will be carry 20% of the total course mark.

The assignment will be set and assessed by the educators who are teaching the subject and moderated by the MATSEC Markers’ Panel. This effort should be spread out over the length of the course. A list of sample topics is included in Section 5. All marks are to be submitted to the MATSEC support unit by the date stipulated by the unit.

**Private candidates** are to submit the assignment to the MATSEC support unit, for assessment by the Markers’ panel, by the date stipulated by the MATSEC support unit. Candidates may be called for an interview regarding their work.

**Note regarding use of calculators:**

Calculators may NOT be used in any part of this examination.

**3. SYLLABUS**

**Module 1: Basic Computing Concepts**

**Objectives**

Students should be able to

- Understand the basic aspects of different computer systems
- Understand the need for different computer systems used for different applications
- Understand the social and ethical implications of the use of computers.

**Content**

*Basics of computing:*

Introduction to:

- Hardware
- Software
- Data & information

Identification of different Hardware and their application in the real world.  
Candidates should also be aware about different types / categories of software as used in the real world e.g. by category:

- Process control: e.g. controlling machines in factories
- Office automation: e.g. Spreadsheet, Word processor
- Industrial & Commercial: e.g. CAD, CAM, Architectural
- Multimedia: A/V editing, Graphics Design, Authoring tools

*Evolution of computer systems*

Classifications of computers:

- Mainframe Computers
- Mini Computers
- Micro Computers

Generations of Computers:

- Candidates should be able to differentiate between generations of computers and they should provide, at least, one example of each generation.  
Characteristics should only be outlined.

*Components of a Computer System in details:*

Input & Output Hardware Devices:

- Different types of input devices and their use

Storage Hardware Devices:

- Diff Storage Hardware devices, Serial/Sequential Storage Devices, Direct Access Storage devices.

Von Neumann Machine architecture

Definition of a program, Application software, Applications packages, System software.

Algorithm:

- Definition of an algorithm, the importance of an algorithm to explain and breakdown a problem when designing programs.
- Introduction to different representations of an algorithm e.g. textual, diagrammatic: pseudo code, flowcharts.
- Introduction to control and data structure of an algorithm.

*E-Learning:*

Electronic Learning as a complimentary form of learning:

- E-learning vs. face-to-face learning style: pros and cons
- Virtual Learning Environments: Candidates should be aware that different VLE's exist e.g. WebCT, BlackBoard, Moodle, etc.

Candidates are not expected to know properties of different VLE's, however, typical content found in a VLE is expected to be known. Differences between open source and commercial source applications.

*Electronic Activities:*

- E-mail: e.g. send plain text messages, attaching files.
- E-Commerce: e.g. Business to consumer, purchasing on-line
- E-Business e.g. Business to Business, EFT
- Remote Access: e.g. Working remotely, logging on remotely

An overview of the above electronic activities is examined. Comparisons including advantages and disadvantages of the above when compared with traditional methods are also expected.

*Social Implication of Computing*

- Computer Crime
- Privacy of Data
- Data Protection Act overview, with reference to the Maltese legislation.
- Web 2.0
- Plagiarism

*Data Representations*

- Decimal, Binary, Hexadecimal number systems
- Converting from one base to another of the above
- Binary two's Complement & Sign and Magnitude representations, range.
- 8421 Binary Coded Decimal
- The use of fixed point number representation to represent fractions (binary only)
- ASCII & Unicode as standard codes.

*Computer Language Translation*

Source Code, Translation including differences between Compilers, Interpreters, Assemblers. Machine Object Code. Pseudo Machine. Description and applications of various compilers & techniques should be overviewed. Details and functioning of these translators should not be tackled.

**Module 2: Digital Logic**

**Objectives**

Students should be able to

- understand the basics behind binary logic
- make use, understand and draw truth tables for logic expressions
- draw logic circuits from Boolean expressions
- apply laws of Boolean algebra and/or Karnaugh maps to simplify a Boolean expression
- design a combinational logic circuit using a simple practical application

**Content**

<i>Logic Gates</i>	<p>Binary logic as a mathematical way of manipulating and processing binary information                      Basic logic operators: AND NOT OR NAND NOR XOR XNOR                      Logic gates, truth tables and digital circuit symbols to represent simple logic solutions                      The use of NAND and NOR as universal gates                      Drawing of logic circuits from Boolean expressions</p>
<i>Boolean Algebra</i>	<p>Basic theorems and properties of Boolean algebra                      Equivalence, contradictions and tautologies                      The following list of laws will be assumed:</p> <ol style="list-style-type: none"> <li>1. Commutative laws.                          (a) <math>X + Y = Y + X</math>; (b) <math>X \cdot Y = Y \cdot X</math></li> <li>2. Associative laws.                          (a) <math>X + (Y + Z) = (X + Y) + Z</math>; (b) <math>X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z</math></li> <li>3. Distributive laws.                          (a) <math>X \cdot (Y + Z) = X \cdot Y + X \cdot Z</math>; (b) <math>X + Y \cdot Z = (X + Y) \cdot (X + Z)</math></li> <li>4. De Morgans laws.                          (a) <math>\overline{(X + Y)} = \bar{X} \cdot \bar{Y}</math>; (b) <math>\overline{(X \cdot Y)} = \bar{X} + \bar{Y}</math></li> <li>5. Laws of absorption.                          (a) <math>X + X \cdot Y = X</math>; (b) <math>X \cdot (X + Y) = X</math></li> <li>6. Double complement law.  <math display="block">\overline{\bar{X}} = X</math></li> <li>7. Laws of tautology.                          (a) <math>X + X = X</math>; (b) <math>X \cdot X = X</math>                          (c) <math>X + \bar{X} = 1</math>; (d) <math>X \cdot \bar{X} = 0</math>                          (e) <math>X + 1 = 1</math>; (f) <math>X \cdot 1 = X</math>                          (g) <math>X + 0 = X</math>; (h) <math>X \cdot 0 = 0</math></li> </ol>
<i>Karnaugh Maps</i>	<p>The use of Karnaugh maps to simplify Boolean expressions.                      Only Karnaugh maps using up to three variables will be considered.</p>
<i>Logic Circuits</i>	<p>Design a simplified logic circuit including the Half Adder and Full Adder from first principles.</p>

### Module 3: Computer Architecture and Assembly Language

#### Objectives

Students should be able to

- gain a good understanding of the main components making up the computer system
- understand the function of the different components making up the system
- have a clear understanding of how the different system components are connected together and how they work to give the required output
- Understand some basic assembly instructions

#### Content

<i>Overview of the Organization of a Computer System</i>	Main PC components <ul style="list-style-type: none"> <li>- CPU</li> <li>- Main Memory</li> <li>- I/O Subsystem</li> <li>- The System clock</li> </ul>
<i>The System Bus</i>	System bus as a means of communication between components <ul style="list-style-type: none"> <li>- Address Bus</li> <li>- Data Bus</li> <li>- Control Bus</li> </ul> Buses size consideration and connection to various computer components
<i>Memory</i>	RAM The characteristics and application of RAM <ul style="list-style-type: none"> <li>- Dynamic RAM</li> <li>- Static RAM</li> <li>- Cache Memory</li> </ul> ROM The characteristics and application of ROM <ul style="list-style-type: none"> <li>- ROM</li> <li>- EPROM</li> <li>- PROM</li> <li>- EEPROM</li> </ul>
<i>CPU</i>	CPU model and overview of main components The fetch, decode and execute cycle with relevance to the Instruction Pointer, Current Instruction Register, Memory Address Register and Memory Data Register.
<i>Registers</i>	The purpose and use of special internal registers in the functioning of the CPU including <ul style="list-style-type: none"> <li>- Data registers (Consider the four GP registers AX, BX, CX, DX)</li> <li>- Stack Pointer and the stack with a description of PUSH and POP</li> <li>- Status or flag register considering the zero and carry flag as examples.</li> </ul>
<i>Assembly Language Instructions</i>	Instruction sets Instruction format – opcode & operand Mnemonic representation of opcode
<i>Assembly Language Instructions</i>	Only a minimal set of Instructions need to be considered: MOV, ADD, SUB, INC, CMP, JG, JL, JE, PUSH and POP
<i>Addressing Modes</i>	Register – Ex SUB AX,BX Immediate – Ex MOV AX,000FH Direct – Ex MOV BX,[2000H]
<i>Assembler</i>	<i>Note students will not be asked to write programs in assembly language</i> The assembly process: assembling, linking, and loading, source and object codes.

## Module 4: Operating Systems

### Objectives

Students should be able to

- describe different operating systems and their function
- outline their interaction
- develop a basic understanding of how operating systems manage memory and files

### Content

<i>Operating Systems</i>	Choice of operating system depends on the type of application software to be used <ul style="list-style-type: none"><li>- Batch</li><li>- Online</li><li>- Multi-access</li><li>- Real time</li></ul>
	Process Control <ul style="list-style-type: none"><li>- Process management</li><li>- Scheduling<ul style="list-style-type: none"><li>- Round Robin</li><li>- Priority</li></ul></li></ul>
<i>Memory Management</i>	<ul style="list-style-type: none"><li>- Virtual memory and Cache Memory</li><li>- Memory store protection (to prevent processes from accessing storage allocated to other jobs)</li></ul>
<i>File Management</i>	File organization <ul style="list-style-type: none"><li>- Creating and accessing files</li></ul>
	Protection of files <ul style="list-style-type: none"><li>- Facilities against unauthorised access<ul style="list-style-type: none"><li>- User ID and password</li><li>- User home directory</li><li>- File access restrictions (system administrator)</li></ul></li><li>- File attributes</li><li>- Hardware failure</li></ul>

## Module 5: Networking and Communications

### Objectives

Students should be able to

- understand the basics of transmission methods in communication
- distinguish between different categories of networks
- appreciate the purpose of a protocol in communication
- appreciate the wide range Internet related technical terms and Internet applications

### Content

<i>Point-to-point Connections</i>	<p>Basics of communications</p> <ul style="list-style-type: none"> <li>- simplex</li> <li>- half duplex</li> <li>- full duplex</li> </ul> <p>Analogue vs. digital signals Bandwidth and baud rate Modems Modulation</p> <ul style="list-style-type: none"> <li>- amplitude</li> <li>- frequency</li> </ul> <p>Demodulation Transmission media</p> <ul style="list-style-type: none"> <li>- cabling (twisted-pair, coaxial, optical fibre)</li> <li>- satellites</li> </ul> <p>Noise</p>
<i>Computer Networks</i>	<p>Definition and classification of a computer network</p> <ul style="list-style-type: none"> <li>- LAN</li> <li>- MAN</li> <li>- WAN</li> </ul>
<i>Network Topologies</i>	<p>The benefits and drawbacks of basic network topologies</p> <ul style="list-style-type: none"> <li>- Bus</li> <li>- Ring</li> <li>- Star</li> <li>- Mesh</li> </ul>
<i>Switching Techniques</i>	<p>The difference between various switching techniques and their application</p> <ul style="list-style-type: none"> <li>- Circuit-switching</li> <li>- Message-switching</li> <li>- Packet-switching</li> </ul>
<i>Communication Protocols</i>	<p>The need for standard protocols</p> <ul style="list-style-type: none"> <li>- The 7 layer OSI model and the IP model</li> </ul> <p>Transmission errors and methods to overcome them</p> <ul style="list-style-type: none"> <li>- Parity (single and block) checking</li> </ul>
<i>International Network Standards</i>	<p>The dotted decimal notation form of IP addressing. Internet domain naming using DNS to assign a host name to every device on the Internet corresponding to the IP address.</p>
<i>Integration of Internet and WWW-related Terminology</i>	<p>Internet service provider, web server, web client, web browser, plug-in, search engine, home page, Uniform Resource Location (URL), bookmarks, ftp, download, upload, remote login, user ID, password, Hypertext Markup Language (HTML), hypermedia, authoring tools, ISDN, ADSL, Proxy Servers, Hubs and routers (<i>only a broad knowledge is expected from the students</i>)</p>

## Module 6: High Level Language Programming and Systems Design

### Objectives

Students should be able to

- Understand the reasons and benefits of system development through sound analysis and design principles
- Gain a basic feel for the application of the above principles by being brought into contact with modern practical commercial software development approaches through a top-level outline and feature/scope comparison of two software development methodology standards known as Structured Systems Analysis and Design Method (SSADM) and the Unified Modelling Language (UML)
- Identify different programming paradigms including imperative, declarative, functional and object-oriented and gain a good understanding of the object oriented and imperative paradigms
- Have a good understanding of the fundamental concepts of object oriented programming including objects and classes, data encapsulation, inheritance and polymorphism

### Content

<i>High Level Languages</i>	<p>Basic Knowledge of different paradigms, specifically:</p> <ul style="list-style-type: none"> <li>- Imperative</li> <li>- Object-Oriented</li> <li>- Formal</li> </ul> <p><i>Only a comparison of non-context free and context free languages from a formal point-of-view should be tackled</i></p> <p>Object-Oriented Programming Characteristics</p> <ul style="list-style-type: none"> <li>- Encapsulation</li> <li>- Inheritance</li> <li>- Polymorphism</li> </ul>
<i>Data Types</i>	<p>Standard Types</p> <ul style="list-style-type: none"> <li>- Numeric types and their ranges</li> <li>- Character (char) and String types</li> <li>- Boolean types</li> <li>- Other classes as types</li> </ul> <p>Constants</p> <p>Variables</p> <ul style="list-style-type: none"> <li>- Scope and visibility of variables</li> </ul>
<i>Conditional</i>	Conditional structure statement
<i>Looping Structures</i>	<p>Pre-tested loops</p> <p>Post-tested loops</p> <p>Nested loops</p>
<i>Methods and classes</i>	<p>The Java API (to be covered from point of view of usage and not content)</p> <p>Class vs. Object</p> <p>Static classes</p>
<i>Data structures and algorithms</i>	Purpose of data structures
<i>Stacks</i>	Basic knowledge of the concept of a stack as a data structure
<i>Linear Lists</i>	Basic knowledge of the concept of lists and queues as data structures
<i>Other Structures</i>	<p>Arrays – single and multi-dimensional</p> <ul style="list-style-type: none"> <li>- Creating an array</li> <li>- Filling in an array with data</li> <li>- Displaying data from an array</li> </ul>
<i>Sorting Algorithms</i>	<p>Basic knowledge of sorting algorithms in general with special reference to</p> <ul style="list-style-type: none"> <li>- Insertion sort</li> <li>- Bubble sort</li> </ul>
<i>Searching Algorithms</i>	Linear Search

<i>Files</i>	Text and Random files <ul style="list-style-type: none"><li>- Text files – Creating, adding data and displaying contents</li></ul> File Operations <ul style="list-style-type: none"><li>- Writing to a file</li><li>- Reading from a file</li></ul>
<i>Software systems</i>	The importance of thinking before doing. A software development life cycle (simplistic “Waterfall”).
<i>Domain research</i>	The basic system domains (scientific, business, and control). Traditional and electronic sources, current system observations, surveys and questionnaires, user/client meetings and interviews.
<i>Requirements establishment</i>	What are system requirements? Bridging the gap between the technical and the non-technical (i.e. between developer and user/client). Systems requirements coverage (Processing, Data, and Input-Output/GUI).
<i>Functional analysis</i>	Using data flow analysis to describe system function (Data Flow Diagrams - DFDs) as the principle analysis tool.
<i>System design</i>	Overview of Design aspects in the form of <ul style="list-style-type: none"><li>- Interface</li><li>- Data</li></ul> Modular system structuring. Top-down and bottom-up design approaches. The use of pseudo-code and/or flowcharts.
<i>Implementation</i>	Programming language selection issues. User guides and in-code comments. Input validation issues. Cross-over techniques: <ul style="list-style-type: none"><li>- Direct</li><li>- Parallel running</li><li>- Phased transition</li></ul>
<i>Fundamentals of testing</i>	Differences of Alpha and Beta testing. I/O testing in the form of validation of inputs and outputs through the creation of test-data within testing ranges. Comparison between black and white box testing concepts.
<i>Maintenance</i>	Outline of the three main categories of maintenance: <ul style="list-style-type: none"><li>- Adaptive</li><li>- Corrective</li><li>- Perfective</li></ul>
<i>Standard methodologies</i>	A top-level overview of both SSADM and UML as a framework within which software development can take place.

## Module 7: Databases & Database Management Systems

### Objectives

Candidates should be able to:

- understand the basic structure, function and importance of database management systems (DBMS)
- appreciate the importance of relational databases over traditional file systems
- understand the logical structure and design of a relational database

### Content

The structure and functions of database management systems (DBMS) including:

- data dictionary
- data manipulation language (DML)
- data description language (DDL)
- query language

Database Administrator:

- The responsibilities of a database administrator
- The role of the DBA in the design and maintenance of the database.

Database models:

- The two main type of database models namely:
- Relational models
  - Object-Oriented Model.

Databases vs. Flat File Systems

The advantages and disadvantages of databases over traditional file systems including: improved data consistency and portability, control over data redundancy, greater security, increased maintenance.

Relational Databases:

The nature and logical structure of a relational database as a set of tables linked together using common fields.

The purpose of primary, secondary and foreign keys, fields, records. Attributes & tuples. Compound Keys.

Normalising relational databases:

Candidates must appreciate that the great advantage of using Relational database is due to the normalization process which is applied on it.

Candidates are ***not*** expected to normalize databases but understand the ***concept*** behind normalization to reduce duplicate data

### LIST OF ACRONYMS

ADSL	- Asymmetric Digital Subscriber Line
ATM	- Asynchronous Transfer Mode
BNF	- Backus Naur Form
DMA	- Direct Memory Access
ROM	- Read Only Memory
EEPROM	- Electrically Erasable Programmable ROM
EPROM	- Erasable Programmable ROM
FTP	- File Transfer Protocol
HDSL	- High bit-rate Digital Subscriber Line

- ISDN - Integrated Services Digital Network
- LAN - Local Area Network
- MAN - Metropolitan Area Network
- PROM - Programmable ROM
- USB - Universal Serial Bus
- WAN - Wide Area Network

## 4. RECOMMENDED TEXTS

*(Students are urged to look for the latest editions, ISBNs will therefore vary accordingly)*

### 4.1. Student's basic text book

Heathcote, P.M., Langfield S., *A-Level Computing*, Payne-Gallway Publishers.

### 4.2. Other recommended sources

Brookshear, J.G., *Computer Science – An Overview*, Addison Wesley.

David, J.B., Kolling, M., *Objects First with Java*, Prentice Hall.

Wu, C.T., *An Introduction to Object-Oriented Programming with Java*, McGraw-Hill.

The use of the Internet, in the form of online documentation and reference sources, is strongly recommended.

### 4.3. Recommended reference books

<i>Computer Architecture and Assembly</i>	Abel, P., <i>IBM PC Assembly Language and Programming</i> . Prentice Hall. Kleitz, W., <i>Digital and Microprocessor Fundamentals</i> . Prentice Hall. Uffenbeck, J., <i>The 80x86 Family: Design, Programming, and Interfacing</i> . Williams, R., <i>Computer Systems Architecture</i> .
<i>Data Structures and Algorithms</i>	Carrano F. M., Prichard J. J., <i>Data Abstraction and Problem Solving with C++: Walls and Mirrors</i> . Addison Wesley.
<i>Databases and SQL</i>	Whitehorn M., Marklyn B., <i>Inside Relational Databases</i> . Springer-Verlag UK. Taylor, A. G., <i>SQL For Dummies</i> . John Wiley & Sons Inc.
<i>Digital Logic</i>	Morris, M., <i>Digital Design</i> . Prentice Hall.
<i>Networking and Communications</i>	Hodson, P., <i>Local Area Networks</i> . Continuum. Stallings, W., <i>Data and Computer Communications</i> . Halsall, F., <i>Computer Networking and the Internet</i> .
<i>Operating Systems</i>	Ritchie, C., <i>Operating Systems. Incorporating Unix and Windows</i> . Continuum. Silberschatz, A., et al., <i>Operating System Concepts</i> Tanenbaum, A.S., Woodhull, A.S., <i>Operating Systems Design and Implementation</i> , Prentice Hall Software Series.
<i>Project Management</i>	Heathcote, P.M., <i>Tackling Computer Projects</i> . Payne-Gallway Publishers.
<i>Systems Analysis and Design</i>	Kendall, J. E., Kendall E. K., <i>Systems Analysis and Design</i> . Prentice Hall. Lejk, M., Deeks, D., <i>Systems Analysis Techniques</i> , Addison Wesley.
<i>Java</i>	Deitel, P.J., Deitel, H.M., <i>Java, How to Program</i> , Prentice Hall. Schildt, H., <i>Java: A Beginner's Guide</i> , Osborne McGraw-Hill. Schildt, H., <i>Java: J2SE (Osborne Complete Reference S.)</i> , McGraw-Hill.
<i>Other</i>	British Computer Society, <i>Glossary of Computer Terms</i> , Addison-Wesley.

## 5. FURTHER INFORMATION REGARDING THE ASSIGNMENTS

### 5.1 Suggested Areas for Assignments

**It is important**, that all assignments be fully documented, with clearly defined explanations of any decisions taken and approaches adopted, together with a description of possible fields of application for the object of the assignment.

*The following are some generic examples of areas to which programming assignments can be related. However, it should be noted that teachers are free to choose any sort of programming example from any field that best suits their needs and expectations.*

***Mathematical type of problems:***

Finding statistical mean, root mean square, finding the maximum and minimum of a set of numbers, compound interest, depreciation.

***Iterative types of problems:***

Factorials. Simple permutations. Combinations. Problems involving repetitive tests with loops and nested loops.

***Character graphics problems:***

Area filling with characters. Cursor movement. Problems involving text screen handling. Use of extended character set. String manipulation problems: Abbreviations case conversion, simple string sorting. Simple string searches. Use of string arrays.

***Mapping problems:***

Mapping structures onto strings. Mapping bit information into binary numbers e.g. maze representation in binary entries. Simple encryption problems.

***Data management problems:***

File handling utilities and environments. Database construction and management. Operations on data (sorting, searching, shifting, etc.). Stock handling programs. Simple statistical analysis.

**Note:** This assignment should include a brief description of the functionality and structure of the program. The program should be well-structured, well documented, and evidence of adequate testing should be documented.

### 5.2 Accredited Schools

Schools presenting candidates for this examination must normally offer full-time courses in Computing and must be accredited by the Maltese education authorities.

It is the responsibility of schools presenting candidates for this examination to ensure that they are properly equipped with the appropriate hardware equipment and software packages for any project work set for the candidates. No concession for candidates lacking the right tools and equipment will be made by the MATSEC Board.

### 5.3 Assessors

The teachers authorised to act as assessors of the project will be appointed by the University. Any authorised assessor reserves the right to interview any candidate of his/her choice regarding the content of the candidate's submitted assignment.