A Level OCR Computer Science

Unit	YEAR 12
PAPER 1	**Learning Objective**
Components of a Computer	
Structure and Function of the Processor	The purpose and function of the core components of a processor.
	The role and components of the ALU.
	The purpose and function or registers within the processor, including the PC, accumulator, MAR, MDR and CIR.
	The purpose, function and role of the data, address and control buses in the processor.
	How assembly language makes use of registers, and how data and addresses are transferred between registers.
	The purpose and stages within the FDE cycle.
	How and when the registers are used within this cycle, and how and where data and addresses are transmitted to/from in each part of this cycle.
	How the performance of the CPU can be affected by many factors.
	How and why the performance is affected by the clock speed, the number of cores, and the size and speed of the cache.
	The Von Neumann and Harvard architectures, the different approaches the architectures take to storing instructions and data in memory, and the benefits of each approach.
	Candidates will not be asked about specific aspects of “contemporary processor architecture” unless explicitly named in the specification. They may, however, be asked to show an awareness of how contemporary processors differ from a pure Von Neumann architecture in more open questions.
	The differences between the CISC and RISC processors, and the key features and benefits of each.
	The relative benefits of each architecture.
	What is meant by a parallel system and the benefits and limitations of parallel processing.

Types of Processor	That parallel processing can be achieved through different methods (i.e. multiple processors in the same computer or distributed, or multiple cores in a CPU or GPU).
	The benefits of a multicore system in terms of parallel processing and running multiple programs at the same time.
	The purpose of GPUs and what applications they are used for (Candidates need to understand how GPUs are used to aid graphics, but also other applications; for example, their use in modelling, data mining, etc.).
	The benefits of using GPUs and why they are suited to certain tasks (specialist instructions, multiple cores and simd processing).
Input, Output and Storage	A range of input, output and storage devices.
	Candidates do not need to understand how the input and output devices work, but must be able to recommend appropriate devices for specific situations and be able to justify choices made.
	That there are different types of storage device; the characteristics of each type (magnetic, optical and flash) and the benefits and drawbacks of each (Candidates need to be able to recommend an appropriate type of device for a given situation and justify the choice).
	The purpose of ROM and RAM within a computer system, their characteristics, and the role they play in the running of a range of different computers, e.g. mobile devices, embedded systems etc.
	Why there is a need for virtual storage, how virtual storage works, and the benefits and drawbacks of using virtual storage. Virtual storage would be that which may appear to be local but is physically located elsewhere on the network/remotely/in the cloud.
Systems Software	
	Why an operating system is required, along with the different tasks it performs within a computer system (e.g. resource management, file management, interrupt handling, security, providing a platform for software to run, providing a user interface and providing utilities).
	How operating systems manage memory (Candidates need to understand the need for, purpose and function of paging to divide memory into usable fixed-size pages and how this aids in the transfer of memory for example virtual memory).

Operating Systems Software	What is meant by segmentation, and how memory is divided into segments to allow access to memory.
	What is meant by virtual memory, and why this is needed in a computer system.
	How paging is used in virtual memory, and the benefits and drawbacks of having and using virtual memory in a computer system.
	The purpose of interrupts within a computer system, why an interrupt might be generated, and what happens within the CPU and memory in order to call an interrupt service routine.
	The need for scheduling of tasks by an operating system and the benefits that scheduling brings.
	That there are different scheduling algorithms, with each having benefits and drawbacks for tasks with specific characteristics.
	How the following scheduling algorithms work: round robin, first-come first-served, multi-level feedback queue, shortest job first, and shortest remaining time.
	The different (and often overlapping) classifications of operating systems (distributed, embedded, multi-tasking, multi-user and real time), including the key features of each (Candidates should be able to recommend (and justify) a type of operating system for a given scenario).
	The role of the BIOS in a computer system, and the steps that the BIOS goes through to start a computer.
	What is meant by 'device drivers' and why they are needed for communication between hardware and the operating system.
	To describe what is meant by a virtual machine, how they can be used to execute intermediate code, how they can be used to run a software driven machine inside a physical machine, and the benefits and drawbacks of each approach.
	The purpose of applications (Candidates should have knowledge and experience of a range of different application software (for example, database, word processor, web browser, graphics manipulation etc.).
	To recommend the use of specific and generic applications for given scenarios, justifying their use and function(s) for a scenario.
	The purpose and role of utility software in a computer system.

Applications Generation	A range of utility software (e.g. disk defragmentation, file management, device driver, system cleanup, security etc.).
	To explain the differences between open and closed source software, and the benefits and drawbacks to creator and user of each of the licensing models, and to be able to recommend which is used (with justification) for a specific scenario.
	The need for translators when writing programs.
	The differences in operation of interpreters and compilers; from these, candidates need to be able to assess the benefits and drawbacks of using each type, and recommend with justification which should be used in a specific scenario.
	The role of an assembler and how it differs from interpreters and compilers.
	That there are a number of stages involved in compilation.
	How lexical analysis works and how the code is converted into tokens with the removal of unnecessary elements (e.g. comments and whitespace).
	How syntax errors are identified and reported at the end of the syntax analysis.
	How the abstract syntax tree will be fed into the next stage of code generation, and that the object code is then created.
	Why optimisation is important and how the results of lexical analysis feed into syntax analysis, and how the tokens are checked to ensure they meet the during (and after) code generation.
	What code libraries are, how they are used, and the benefits and drawbacks from using libraries.
Software Development	
	The different models that can be followed to produce a program (explicitly the waterfall lifecycle, agile methodology, extreme programming, the spiral model, and rapid application development).
	The tasks, processes, benefits and drawbacks of each model, and the similarities and differences between each.
	Where each model is most suitable to use, and to be able to justify the use in a situation.
	To write algorithms using flow charts, pseudocode and/ or program code.
	To follow the code as shown in the OCR pseudocode guide (Candidates are not expected to write code in this).

Software Development	Candidate's code is not expected to be syntactically correct, but must use appropriate code structures.
	Using black box testing, white box testing, alpha testing and beta testing whilst producing their own programs.
	How each testing strategy can be used in a situation, and the benefits and drawbacks of each method, applying this to a given situation to recommend appropriate testing strategies.
	Using suitable test data to test their own programs.
	The use of test data and applying this to a given program.
	How dry runs can be used in the development and testing of programs, and being able to use dry runs to test given code.
	The need for and importance of end user feedback.
	That there are a variety of types of programming paradigms, such as procedural, OOP, low-level, and that each has its strengths and weaknesses in specific scenarios, topics or areas.
	Candidates need to have knowledge and experience of using a procedural programming language; for example, Python, vb.net etc.
	To read, trace, amend and write procedural program code.
	The purpose and need for assembly language. Candidates need to be familiar with the instructions given in Appendix 5d, and they should be able to read, write, trace and amend programs written in the Little Man Computer Language.
	Addressing, which should be integrated with assembly language.
	Using immediate, direct, indirect and indexed addressing in the writing, reading and tracing of programs written in assembly language.
	Object-oriented code (as specified in the pseudocode guide). Candidates need to have an understanding of classes, objects, attributes and methods, and need to understand the difference between private and public attributes and methods.
	Encapsulation and the use of get and set methods to access private attributes.
	The purpose and principles of inheritance.
	Polymorphism and how it can be used within a program.
	To read, trace, amend and write code that makes use of these object-oriented features.

Types of Programming Language	How to control the flow of a program (sequence, iteration and selection).
	The purpose and function of both variables and constants, and be able to read, trace and write code that makes use of both variables and constants.
	The benefits of using constants over variables.
	The role of sub-programs (procedures and functions) in a program, and how these can be used to reduce the amount of code and improve the efficiency.
	The differences between procedures and functions, and being able to read, write and trace programs using both procedures and functions.
	Using a range of arithmetic (+, -, /, *, mod, div) operators, Boolean (and, or, not, ==, >, <, =, >=, <=, !=) operators, and assignment operators (=).
	To read, trace and write programs using arithmetic and Boolean operators. Code in the exam will be written using the OCR pseudocode guide, so candidates need to be able to read and interpret this pseudocode – however, their answers can be in pseudocode, or program code.
	Using a range of string handling functions. Candidates need to be able to read, trace and write program code using and combining string handling techniques (selecting substrings, converting to upper/lowercase, converting between characters and their ascii values). Any functions presented in a question which are not in pseudocode guide, will be specifically introduced.
	Writing programs that write to and read from text files. Candidates' understanding of procedural languages will largely be tested by asking candidates to read/write/trace/amend simple programs.
Exchanging Data	
Compression, Encryption and Hashing	The need for compression, especially when transferring data via the internet.
	The difference between lossy and lossless compression, and the benefits and drawbacks of each type.
	To recommend a type of compression for a given scenario.
	How run-length encoding can reduce the size of a file; for example, with a text file or image.
	How dictionary coding works by substituting entries with a unique code.
	Candidates should have practical experience of using these algorithms with small example files.
	The need for encryption.

	How symmetric and asymmetric encryption work to encrypt and decrypt data.
	The need for and purpose of using hashing algorithms to store data.
	Different uses for hashing, such as the storing of passwords.
Databases	What is meant by a database.
	Basic database terminology, such as fields, records and tables.
	The difference between a flat file and a relational database, and being able to explain the benefits and limitations of each approach.
	Setting up and using both a flat file and relational database.
	What is meant by a primary key, foreign key and secondary key, and how each are used in a database.
	To produce and follow entity relationship (er) diagrams which include 1:1, 1:m and m:m relationships.
	To identify how tables should be linked.
	A range of methods for capturing data (such as forms, ocr, omr and sensors), selecting data (such as query by example and sql), managing data (such as changing data by manipulating it – e.g. arithmetic functions, adding, editing, deleting the data), and exchanging data (with common formats such as csv, json and xml). Candidates won't be specifically asked about any one of these methods but may be asked to discuss/justify suitable methods as part of a more open question.
	The need to interrogate data within a database.
	The purpose of indexing in a database and the benefits of using indexing to optimise the searching for data.
	Candidates need to have experience of a range of methods for capturing data (such as forms – what do they collect, what do they look like – data mining, where does the data come from, how is it collected and analysed), selecting data (such as how to produce qbes – adding fields, tables, criteria, sorting – selecting through Boolean expressions – and, or, not), managing data (such as changing data by manipulating it – e.g. arithmetic functions – , adding, editing, deleting the data), and exchanging data (such as methods of transferring data – electronic i.e. memory stick, e-mail, and non-electronic e.g. paper based – appropriate formats for the transfer of data and communication mediums to transfer data – such as the structure, is it in a table or a list).

	Using SQL to edit and modify data in a database, and understanding the need for SQL as a standard language.
	To write and follow scripts using the SQL commands listed in Appendix 5d.
	What is meant by referential integrity, and why this is desirable in a database.
	What is meant by transaction processing, and scenarios where transaction processing takes place.
	The problems that arise from transaction processing, and how these can be overcome.
	The acid rules for transaction processing, and why databases should be built to these standards.
	How record locking prevents the overriding of data, and understanding how record locking takes place.
Networks and Web Technologies	
Networks	The definition and purpose of a network.
	The purpose of, and importance of using, protocols.
	To discuss examples of protocols that may be used in a network/the internet (Candidates will not be asked to recall information about any specific protocol).
	The term 'standard', and the purpose and need for standards in a network (or any situation where data is transferred).
	The purpose and benefits of layering protocols, particularly within the TCP/IP stack; candidates need to know the different layers within the TCP/IP stack and the purpose of each.
	How data is transmitted on the internet, and the use of IP addresses and packets in the transfer of data. (Candidates are not expected to be familiar with the OSI model).
	To understand the terms 'LAN' and 'WAN'.
	How the domain name system is used to find the IP address of a URL.
	The purpose, function, benefits and drawbacks of both packet and circuit switching.
	The difference between a client-server and peer-to-peer network.
	The benefits and drawbacks of each type of network, and recommending one for a given scenario.
	That there are a range of security issues and threats involved with networked computers.
	Awareness of threats such as hackers, viruses, unauthorised access, denial of service, spyware, SQL injection, phishing and pharming.

	To know about ways of minimising, or preventing these threats; for example, firewalls, secure passwords, anti-virus, anti-spyware etc.
	To have knowledge of the hardware required to connect to and/or build a network (e.g. modem, router, cable, nic, wireless access points, hub, switch etc).
	The purpose of the hardware (Candidates are not required to understand how they physically work).
Web Technologies	The purpose of HTML, CSS and Javascript.
	Knowledge of when each language/markup would be used, and what its purpose and function is.
	Writing webpages using HTML, CSS and Javascript.
	To recognise the code in Appendix 5d, and be able to read, write, amend and interpret code using HTML, CSS and Javascript.
	The need for compression (when transferring data over a network).
	The difference between lossy and lossless compression, and the benefits and drawbacks of each type.
	To recommend a type of compression for a given scenario.
	How and why search engine results are indexed, and how PageRank ranks these results.
	How PageRank works at a high level (Candidates are not expected to be able to code the algorithm).
	The difference between server and client side processing, and an awareness of examples (for example Javascript code vs PHP code) of processing on both sides.
	The benefits and drawbacks of both types of processing.
Data Types, Data Structures and Algorithms	
	Programming data types, such as integer, real, Boolean, character, string etc.
	Choosing appropriate data types for a situation or given data.
	Programming solutions using these data types.
	How to convert from one data type to another (casting).
	How and why computers store data as binary, and that a binary number can have a variety of different interpretations depending on what is being stored (e.g. numeric, text, image, sound).
	To convert positive whole numbers to binary and from binary to denary.
	How to store negative numbers using sign and magnitude and two’s complement.

Data Types	To convert denary numbers to sign and magnitude and two's complement – and vice-versa.
	To perform addition and subtraction on integer binary numbers (These numbers could be positive or negative using two's complement representation.)
	The purpose and potential uses of hexadecimal; for example, where and why they are used instead of binary, and the benefits of using hexadecimal over alternatives such as binary.
	To convert denary numbers to hexadecimal and vice-versa, and from binary to hexadecimal and vice-versa.
	How (positive and negative) real numbers are represented in a binary floating-point representation, and converting between a denary number and a real binary number (NB: The representation used for the exam is the mantissa and exponent, both represented using two's complement).
	The need for normalised floating point numbers.
	To normalise a floating point number.
	How characters are represented in binary.
	The need for a character set and how a computer makes use of a character set.
	The ASCII and UNICODE character sets, and the ability to explain the differences between these and the benefits of each.
	To use a character set, or part of a character set, to translate characters into binary and vice-versa (Candidates are not expected to memorise any values in a character set).
	To perform addition and subtraction floating point arithmetic, including addition and subtraction of both positive and negative numbers.
	To perform right and left logical shifts.
	The effect of right and left shifts on binary numbers.
	The purpose of using masks with bitwise operators, and experience of applying masks using and, or, and xor.
	To describe what is meant by arrays (up to 3 dimensions), records, lists and tuples.
	To be able to recognise when they can be used, and incorporate them in programs to store data.
	The purpose and use of a record structure to store data of different data types in a program.
	Using records to store, search, manipulate and retrieve data.
	The purpose and use of a list to store data in a program.

Data Structures	Using lists to store, search, manipulate and retrieve data.
	The purpose and use of tuples to store data in a program.
	Using tuples to store, search, manipulate and retrieve data.
	The behaviour of stacks and queues (i.e. lifo and fifo).
	The behaviour of linked-lists, graphs, stacks, queues, trees, binary search trees and hash tables.
	To be aware of how the aforementioned data structures can be implemented (Candidates should have general understanding of these principles that can be applied to a given scenario, rather than trying to memorise code patterns).
	Implementing these structures in a variety of contexts; for example, through a procedural program, through a different data structure, and through an object-oriented approach.
	To read, trace and write code to implement features of these data structures (Candidates should have a general understanding backed up with practice implementing them, rather than trying to memorise code patterns).
Boolean Algebra	Be familiar with and, or, not ,and xor, the logic of each Boolean operator, and the truth tables.
	To construct logic gate diagrams from a Boolean expression and vice versa.
	To construct truth tables from Boolean expressions and logic gate diagrams.
	Boolean expressions can be simplified, and candidates should have experience of simplifying expressions using Karnaugh maps.
	To create, complete and interpret Karnaugh maps to simplify Boolean expressions.
	Applying the given De Morgan’s Laws to a Boolean statement.
	Manipulating and simplifying Boolean statements using these rules of distribution, commutation, association and double negation.
	The purpose and principles of D type flip flops, and how and where they are used in a computer (Candidates should be able to recognise how they can be triggered by a clock pulse).
	Candidates are not expected to memorise the logic gates that make up a D type flip flop.

	The purpose and function of an adder circuit, and the difference between a half and full adder (Candidates should be able to recognise and draw the logic gates and truth tables for full and half adders).
Legal, Moral, Ethical and Cultural Issues	
Computer Related Legislation	The need for, and purpose of, laws relating to the use of computers.
	Candidates should be familiar with the purpose and role of the Data Protection Act.
	Candidates will need to understand the different rules that are within the DPA and how these impact the use of computers and the storage of data by organisations. This should include what organisations can and cannot do.
	The purpose and principles of the Computer Misuse Act, including the actions that it prohibits.
	The purpose and principles of the Copyright and Patents Act, including the actions that it prohibits.
	The purpose and principles of the Regulation of Investigatory Powers Act, and what this allows in interception and monitoring of electronic communication.
	How the regulations impact organisations and the use of computers and electronic communication.
Moral and Ethical Issues	What is meant by moral, social, ethical and cultural issues in relation to the use of computers.
	How the use of computers, and the increasing use of computers in the work force, has moral, social, ethical and cultural implications and risks for a variety of people, such as the employees, employers, society and organisations.
	How the use of computers to make decisions automatically has moral, social, ethical and cultural implications and risks for a variety of people such as those people who make the decisions, the people the decisions affect, and the need for additional collection of information to ensure the decisions are accurate and valid.
	How the development of artificial intelligence has moral, social, ethical and cultural impacts on a variety of people.
	How the environmental effects of computers (such as disposal, energy use) have moral, social, ethical and cultural implications.
	How the internet and censorship on the internet has moral, social, ethical and cultural implications.
	The moral, social, ethical and cultural implications of using computers to monitor behaviour (such as CCTV, tracking phone calls, GPS, monitoring emails).

	The moral, social, ethical and cultural implications of using computers to analyse personal information (such as the gathering, storing and analysing of medical records).
	How different cultures impact on the use of and creation of computers and programs; for example, languages make use of different characters, and how this in turn impacts the use of character sets. Some languages read left to right, and others right to left.
	How colours have different meanings in different cultures; for example, red means danger in one culture, and luck in another.
	Candidates need to consider how these will impact the creation of computer applications.
	In order to prepare for this section, we would recommend candidates regularly keep abreast of technological developments in the news.