TAUNTON SCHOOL

Year 12 Syllabus in a nutshell

A Level COMPUTER SCIENCE

Year 12 Syllabus in a nutshell – A Level Computer Science

Computer Science encourages learners to meet the needs of higher education courses in computer science as well as twenty-first century digital employers. It encourages learner to think creatively, through applying practical programming solutions, demonstrating that they are effective uses of technology.

Learners develop computational thinking & programming skills to solve computer science problems. Computer Science will help learners develop a range of skills such as thinking creatively, analytically, logically and critically. They will also be able to appreciate the ethical issues that arise with current and emerging computing technologies.

Syllabus:

https://www.cambridgeinternational.org/programmes-and-qualifications/cambridge-international-as-and-a-level-computer-science-9618/

Text Book:

Computer Science for Cambridge International AS and A Level Coursebook: Sylvia Langfield and Dave Duddell

Year 12

	Autumn	Spring	Summer
Theory	Information Representation, Communication And Internet Technologies, Hardware, Logic Gates And Logic Circuits	Processor Fundamentals, Assembly Language Programming, System Software	Data Security Privacy And Integrity, Ethics And Ownership Database And Data Modelling
Practical	Algorithm design and problem solving	Data types and Structure, Programming and data representations	Software Development

Year 13

	Autumn	Spring	Summer
Theory	Data Representation, Communication And Internet Technologies, Boolean Algebra And Logic Circuits, Artificial Intelligence, Processor And Computer Architecture	System Software, Security, Monitoring And Control Systems	Revision And Exam Preparation
Practical	Algorithms, Recursion, Programming Paradigms, File Processing and Exception Handling, Object Oriented Programming, Low Level Programming, Declarative Programming	Developing Pre Release Solutions	Revision And Exam Preparation

Units In Detail

Information Representation,	• Show understanding of binary magnitudes and the difference • between binary prefixes and decimal prefixes • Show understanding of the basis of different number systems • Perform binary addition and subtraction • Describe practical applications where Binary Coded Decimal (BCD) and Hexadecimal are used • Show understanding of and be able to represent character data in its internal binary form, depending on the character set used • Show understanding of how data for a bitmapped image are encoded • Perform calculations to estimate the file size for a bitmap image • Show understanding of the effects of changing elements of a bitmap image on the image quality and file size • Show understanding of how data for a vector graphic are encoded • Justify the use of a bitmap image or a vector graphic for a given task • Show understanding of how sound is represented and encoded • Show understanding of the impact of changing the sampling rate and resolution • Show understanding of the need for and examples of the use of compression

	• Show understanding of lossy and lossless compression and justify the use of a method in a given situation • Show understanding of how a text file, bitmap image, vector graphic and sound file can be compressed
Communication And Internet Technologies,	• Show understanding of the purpose and benefits of networking devices • Show understanding of the characteristics of a LAN (local area network) and a WAN (wide area network) • Explain the client-server and peer-to-peer models of networked computers • Show understanding of thin-client and thick-client and the differences between them • Show understanding of the bus, star, mesh and hybrid topologies • Show understanding of cloud computing • Show understanding of the differences between and implications of the use of wireless and wired networks • Describe the hardware that is used to support a LAN • Describe the role and function of a router in a network • Show understanding of Ethernet and how collisions are detected and avoided • Show understanding of bit streaming • Show understanding of the differences between the World Wide Web (WWW) and the internet Describe the hardware that is used to support the internet • Explain the use of IP addresses in the transmission of data over the internet • Explain how a Uniform Resource Locator (URL) is used to locate a resource on the World Wide Web (WWW) and the role of the Domain Name Service (DNS)
Hardware,	• Show understanding of the need for input, output, primary memory and secondary (including removable) storage • Show understanding of embedded systems • Describe the principal operations of hardware devices • Show understanding of the use of buffers • Explain the differences between Random Access Memory (RAM) and Read Only Memory (ROM) • Explain the differences between Static RAM (SRAM) and Dynamic RAM (DRAM) • Explain the difference between Programmable ROM (PROM), Erasable Programmable ROM (EPROM) and Electrically Erasable Programmable ROM (EEPROM) • Show an understanding of monitoring and control systems
Logic Gates And Logic Circuits	• Use the following logic gate symbols: NOT AND OR NAND NOR XOR • Understand and define the functions of : NOT, AND, OR, NAND, NOR and XOR (EOR) gates • All gates except the NOT gate will have two inputs only. • Construct the truth table for each of the logic gates above • Construct a logic circuit • Construct a truth table • Construct a logic expression

Processor Fundamentals,	• Show understanding of the basic Von Neumann model for a computer system and the stored program concept • Show understanding of the purpose and role of registers, including the difference between general purpose and special purpose registers • Show understanding of the purpose and roles of the Arithmetic and Logic Unit (ALU), Control Unit (CU) and system clock, Immediate Access Store (IAS) • Show understanding of how data are transferred between various components of the computer system using the address bus, data bus and control bus • Show understanding of how factors contribute to the performance of the computer system • Understand how different ports provide connection to peripheral devices • Describe the stages of the Fetch-Execute (F-E) cycle • Describe and use 'register transfer' notation to describe the F–E cycle • Show understanding of the purpose of interrupts
Assembly Language Programming	• Show understanding of the relationship between assembly language and machine code • Describe the different stages of the assembly process for a two-pass assembler • Apply the two-pass assembler process to a given simple assembly language program • Trace a given simple assembly language program • Show understanding that a set of instructions are grouped • Modes of addressing • Show understanding of and perform binary shifts • Show understanding of how bit manipulation can be used to monitor / control a device

System Software	• Explain why a computer system requires an Operating System (OS)
	• Explain the key management tasks carried out by the Operating System
	• Show understanding of the need for typical utility software provided with an Operating System
	• Show understanding of program libraries
	• Show understanding of the need for:
	  ○ assembler software for the translation of an assembly language program
	  ○ a compiler for the translation of a high-level language program an
	  ○ interpreter for translation and execution of a high-level language program
	• Explain the benefits and drawbacks of using either a compiler or interpreter and justify the use of each
	• Show awareness that high-level language programs may be partially compiled and partially interpreted, such as Java Describe features found in a typical Integrated Development Environment (IDE)

Data Security Privacy And Integrity,	• Explain the difference between the terms security, privacy and integrity of data

	• Show appreciation of the need for both the security of data and the security of the computer system • Describe security measures designed to protect computer systems, ranging from the stand-alone PC to a network of computers • Show understanding of the threats to computer and data security posed by networks and the internet • Describe methods that can be used to restrict the risks posed by threats • Describe security methods designed to protect the security of data • Describe how data validation and data verification help protect the integrity of data • Describe and use methods of data validation Including range check, format check, length check, presence check, existence check, limit check, check digit • Describe and use methods of data verification during data entry and data transfer • Describe and use methods of data verification during data entry and data transfer
Ethics And Ownership	• Show understanding of the need for and purpose of ethics as a computing professional • Show understanding of the need to act ethically and the impact of acting ethically or unethically for a given situation • Show understanding of the need for copyright legislation • Show understanding of the different types of software licencing and justify the use of a licence for a given situation • Show understanding of Artificial Intelligence (AI)
Database And Data Modelling	• Show understanding of the limitations of using a file- based approach for the storage and retrieval of data • Describe the features of a relational database that address the limitations of a file-based approach • Show understanding of and use the terminology associated with a relational database model • Use an entity-relationship (E-R) diagram to document a database design • Show understanding of the normalisation process • Explain why a given set of database tables are, or are not, in 3NF • Produce a normalised database design for a description of a database, a given set of data, or a given set of tables • Show understanding of the features provided by a Database Management System (DBMS) that address the issues of a file based approach • Show understanding of how software tools found within a DBMS are used in practice • Show understanding that DBMS carries out all creation / modification of the database structure using its Data Definition Language (DDL) • Show understanding that the DBMS carries out all queries and maintenance of data using its DML • Show understanding that the industry standard for both DDL and DML is Structured Query Language (SQL)

	• Understand given SQL (DDL) commands and be able to write simple SQL (DDL) commands using a sub-set of commands • Write an SQL script to query or modify data (DML) which are stored in (at most two) database tables
Algorithm design and problem solving	• Show an understanding of abstraction • Describe and use decomposition • Show understanding that an algorithm is a solution to a problem expressed as a sequence of defined steps • Use suitable identifier names for the representation of data used by a problem and represent these using an identifier table • Write pseudocode that contains input, process and output • Write pseudocode using the three basic constructs of sequence, selection and iteration (repetition) • Document a simple algorithm using pseudocode Write pseudocode from: • a structured English description • a flowchart • Describe and use the process of stepwise refinement to express an algorithm to a level of detail from which the task may be programmed • Use logic statements to define parts of an algorithm solution
Data types and Structure,	• Select and use appropriate data types for a problem solution • Show understanding of the purpose of a record structure to hold a set of data of different data types under one identifier • Use the technical terms associated with arrays • Select a suitable data structure (1D or 2D array) to use for a given task • Write pseudocode for 1D and 2D arrays • Write pseudocode to process array data • Show understanding of why files are needed • Write pseudocode to handle text files that consist of one or more lines • Show understanding that an ADT is a collection of data and a set of operations on those data • Show understanding that a stack, queue and linked list are examples of ADTs • Use a stack, queue and linked list to store data • Describe how a queue, stack and linked list can be implemented using arrays

Programming and data representations	• Implement and write pseudocode from a given design presented as either a program flowchart or structured English
	• Write pseudocode statements for:
	  ○ the declaration of variables and constants
	  ○ the assignment of values to variables and constants
	  ○ expressions involving any of the arithmetic or logical operators input from the keyboard and output to the console
	• Use built-in functions and library routines
	• Use pseudocode to write:
	  ○ an 'IF' statement including the 'ELSE' clause and nested IF statements

	○ a 'CASE' structure ○ a 'count-controlled' loop: ○ a 'post-condition' loop ○ a 'pre-condition' loop • Justify why one loop structure may be better suited to solve a problem than the others • Define and use a procedure • Explain where in the construction of an algorithm it would be appropriate to use a procedure • Define and use a function • Explain where in the construction of an algorithm it is appropriate to use a function • Use the terminology associated with procedures and functions • Write efficient pseudocode
Software Development	• Show understanding of the purpose of a development life cycle • Show understanding of the need for different development life cycles depending on the program being developed • Describe the principles, benefits and drawbacks of each type of life cycle • Show understanding of the analysis, design, coding, testing and maintenance stages in the program development life cycle • Use a structure chart to decompose a problem into sub-tasks and express the parameters passed between the various modules/procedures/functions which are part of the algorithm design • Show understanding of the purpose of state-transition diagrams to document an algorithm • Show understanding of ways of exposing and avoiding faults in programs • Locate and identify the different types of errors • Correct identified errors • Show understanding of the methods of testing available and select appropriate data for a given method • Show understanding of the need for a test strategy and test plan and their likely contents • Choose appropriate test data for a test plan • Show understanding of the need for continuing maintenance of a system and the differences between each type of maintenance • Analyse an existing program and make amendments to enhance functionality
Data Representation,	• Show understanding of why user-defined types are necessary • Define and use non-composite types • Define and use composite data types • Choose and design an appropriate user-defined data type for a given problem • Show understanding of the methods of file organisation and select an appropriate method of file organisation and file access for a given problem • Show understanding of methods of file access • Show understanding of hashing algorithms • Describe the format of binary floating-point real numbers

	• Convert binary floating-point real numbers into denary and vice versa • Normalise floating-point numbers • Show understanding of the consequences of a binary representation only being an approximation to the real number it represents (in certain cases) • Show understanding that binary representations can give rise to rounding errors
Communication And Internet Technologies,	• Show understanding of why a protocol is essential for communication between computers • Show understanding of how protocol implementation can be viewed as a stack, where each layer has its own functionality • Show understanding of the TCP / IP protocol suite • Show understanding of protocols (HTTP, FTP, POP3, IMAP, SMTP, BitTorrent) and their purposes • Show understanding of circuit switching • Show understanding of packet switching
Boolean Algebra And Logic Circuits,	• Show understanding of Reduced Instruction Set Computers (RISC) and Complex Instruction Set Computers (CISC) processors • Show understanding of the importance / use of pipelining and registers in RISC processors • Show understanding of the four basic computer architectures • Show understanding of the characteristics of massively parallel computers • Show understanding of the concept of a virtual machine • Produce truth tables for logic circuits including half adders and full adders • Show understanding of a flip-flop (SR, JK) • Show understanding of Boolean algebra • Show understanding of Karnaugh maps (K-map)
Artificial Intelligence,	• Show understanding of how graphs can be used to aid Artificial Intelligence (AI) • Show understanding of how artificial neural networks have helped with machine learning • Show understanding of Deep Learning, Machine Learning and Reinforcement Learning and the reasons for using these methods. • Show understanding of back propagation of errors and regression methods in machine learning •
System Software, Security,	• Show understanding of how an OS can maximise the use of resources • Describe the ways in which the user interface hides the complexities of the hardware from the user • Show understanding of process management • Show understanding of virtual memory, paging and segmentation for memory management • Show understanding of how an interpreter can execute programs without producing a translated version

	• Show understanding of the various stages in the compilation of a program • Show understanding of how the grammar of a language can be expressed using syntax diagrams or Backus-Naur Form (BNF) notation • Show understanding of how Reverse Polish Notation (RPN) can be used to carry out the evaluation of expressions • Show understanding of how encryption works • Show awareness of the Secure Socket Layer (SSL)/ Transport Layer Security (TLS) • Show understanding of digital certification
Algorithms,	• Show understanding of linear and binary searching methods • Show understanding of insertion sort and bubble sort methods • Show understanding of and use Abstract Data Types (ADT) • Show how it is possible for ADTs to be implemented from another ADT • Show understanding that different algorithms which perform the same task can be compared by using criteria (e.g. time taken to complete the task and memory used)
Programming Paradigms,	• Understanding what is meant by a programming paradigm • Show understanding of the characteristics of a number of programming paradigms
Recursion,	• Show understanding of recursion • Show awareness of what a compiler has to do to translate recursive programming code
File Processing and Exception Handling,	• Write code to perform file-processing operations • Show understanding of an exception and the importance of exception handling
Object Oriented Programming,	• Understanding what is meant by a programming paradigm • Show understanding of the characteristics of a number of programming paradigms:
Low Level Programming,	• Understanding what is meant by a programming paradigm • Show understanding of the characteristics of a number of programming paradigms
Declarative Programming	• Understanding what is meant by a programming paradigm • Show understanding of the characteristics of a number of programming paradigms