Qualification Accredited

OCR
Oxford Cambridge and RSA

AS LEVEL
Clarification Guide

COMPUTER SCIENCE

H046
For first teaching in 2015

Subject content clarification

Version 1

www.ocr.org.uk/computerscience

Contents

AS LEVEL COMPUTER SCIENCE

Component 1

1.1. The characteristics of contemporary processors, input, output and storage devices

Components of a computer and their uses		Content clarification	Links to other topics
1.1.1 Structure and function of the processor	a) The Arithmetic and Logic Unit; ALU, Control Unit and Registers (Program Counter; PC, Accumulator; ACC, Memory Address Register; MAR, Memory Data Register; MDR, Current Instruction Register; CIR). Busses: data, address and control: How this relates to assembly language programs. b) The Fetch-Decode-Execute Cycle, including its effect on registers. c) The factors affecting the performance of the CPU, clock speed, number of cores, cache d) Von Neumann, Harvard and contemporary processor architecture	Candidates need to have an understanding of the purpose and function of the core components of a processor. Candidates need to understand the role and components of the ALU. Candidates need to understand the purpose and function of registers within the processor, including the PC, accumulator, MAR, MDR and CIR. Candidates need to understand the purpose, function and role of the data, address and control buses in the processor. Candidates need to understand how assembly language makes use of registers, and how data and addresses are transferred between registers. Candidates need to understand the purpose and stages within the FDE cycle. Candidates need to understand 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. Candidates need to understand how the performance of the CPU can be affected by many factors. Candidates need to understand how and why the performance is affected by the clock speed, the number of cores and the size, speed and cache. Candidates need to have an understanding of the Von Neumann and Harvard architectures. Candidates should be aware of 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. Candidates may, however, be asked to show an awareness of how contemporary processors differ from a pure von Neumann architecture in more open questions.	**1.2.3 Introduction to programming** http://www.teach-ict.com/2016/AS_Computing/OCR_H046/1_1_characteristics_components/111_architecture/von_neumann/home_as_von_neumann.html https://youtu.be/UdHK35N-Kuo http://www.bbc.co.uk/education/guides/zmb9mp3/revision/4 https://youtu.be/OTDTdTYId2g https://youtu.be/Nz2JJXYKWJc
1.1.2 Types of processor	a) The differences between and uses of CISC and RISC processors. b) Multicore and Parallel systems.	Candidates need to understand the differences between the CISC and RISC processors and the key features and benefits of each. Candidates should be aware of the relative benefits of each architecture. Candidates need to understand what is meant by a parallel system and the benefits and limitations of parallel processing. Candidates need to understand that parallel processing can be achieved through different (i.e. multiple processors in the same computer or distributed or multiple cores in a CPU or GPU). Candidates need to understand the benefits of a multicore system in terms of parallel processing and running multiple programs at the same time.	**1.1.1 Structure and function of the processor** https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/ https://youtu.be/BJpMmq9gQE8 http://www.teach-ict.com/as_as_computing/ocr/H447/F453/3_3_3/parallel_processors/miniweb/pg5.htm https://youtu.be/CntADU-4_Gw

Components of a computer and their uses		Content clarification	Links to other topics
1.1.3 Input, output and storage	a) How different input output and storage devices can be applied to the solution of different problems. b) The uses of magnetic, flash and optical storage devices. c) RAM and ROM. d) Virtual storage.	Candidates need to have an understanding of 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. Candidates need to understand that there are different types of storage device. Candidates need to know about the characteristics of each type (magnetic, optical and flash) and understand the benefits and drawbacks of each, and be able to recommend an appropriate type of device for a given situation and justify the choice. Candidates need to understand 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. Candidates need to understand 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.	**1.1.1 Structure and function of the processor** **1.1.2 Types of processor** http://www.computerhope.com/issues/ch001361.htm https://youtu.be/vbFDsSnfLfw https://youtu.be/WjZoIgnayU4

1.2. Software and software development

Types of software and the different methodologies used to develop software		Content clarification	Links to other topics
1.2.1 Operating Systems	a) The need for, function and purpose of operating systems. b) Memory Management (paging, segmentation and virtual memory). c) Interrupts, the role of interrupts and Interrupt Service Routines (ISR), role within the fetch decode execute cycle d) Scheduling: Round Robin, First come first served, Multi-level feedback queues, shortest job first and shortest remaining time. e) Distributed, Embedded, Multi-Tasking, Multi-User and Real Time operating systems. f) BIOS. g) Device drivers. h) Virtual machines, any instance where software is used to take on the function of a machine including, executing intermediate code or running an operating system within another	Candidates need to have an understanding of 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). Candidates need to understand 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. Candidates need to understand what is meant by segmentation and how memory is divided into segments to allow access to memory. Candidates need to understand what is meant by virtual memory and why this is needed in a computer system. Candidates need to understand how paging is used in virtual memory, and the benefits and drawbacks of having and using virtual memory in a computer system. Candidates need to understand the purpose of interrupts within a computer system. Candidates need to understand why an interrupt might be generated, and what happens within CPU and memory in order to call an interrupt service routine. Candidates need to understand the need for scheduling of tasks by an operating system and the benefits that scheduling brings. Candidates need to understand that there are different scheduling algorithms, which each have benefits and drawbacks for tasks with specific characteristics. Candidates need to understand how the following scheduling algorithms work; round robin, first come first served, multi-level feedback queue, shortest job first and shortest remaining time. Candidates need to understand 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. Candidates need to understand the role of the BIOS in a computer system, and the steps that the BIOS goes through to start a computer. Candidates need to understand what is meant by 'device drivers' and why they are needed for communication between hardware and the operating system. Candidates should be able 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.	**1.1.3 Input, output and storage** http://www.teach-ict.com/2016/AS_Computing/OCR_H046/1_2_software/121_operating_systems/purpose_of_os/home_os_purpose.html https://youtu.be/e9klVeFgzMI http://www.studytonight.com/operating-system/cpu-scheduling https://www.tutorialspoint.com/operating_system/os_types.htm http://homepage.cs.uri.edu/faculty/wolfe/book/Readings/Reading07.htm

Types of software and the different methodologies used to develop software		Content clarification	Links to other topics
1.2.2 Applications generation	a) The nature of applications, justifying suitable applications for a specific purpose. b) Utilities. c) Open source vs Closed source. d) Translators: Interpreters, compilers and assemblers.	Candidates need to understand the purpose of applications, and should have knowledge and experience of a range of different application software (for example database, word processor, web browser, graphics manipulation etc.). Candidates should be able to recommend the use of specific and generic applications for given scenarios, justifying their use and function(s) for a scenario. Candidates need to understand the purpose and role of utility software in a computer system. Candidates should be familiar with a range of utility software (e.g. disk defragmentation, file management, device driver, system cleanup, security etc.) Candidates need to be able to explain the differences between open and closed source software, the benefits and drawbacks to creator and user of each of the licensing models, and be able to recommend which is used (with justification) for a specific scenario. Candidates need to understand the need for translators when writing programs. Candidates need to have knowledge of the differences in operation of interpreters and compilers, from these they 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. Candidates need to understand the role of an assembler and how it differs from interpreters and compilers.	**1.2.1 Operating Systems** https://youtu.be/gIzbuiSnuUA https://www.youtube.com/watch?v=_PAsXKddNF4 https://www.itgct.com/whats-the-difference-between-open-source-and-closed-source-software/ https://youtu.be/kPp_XPbr9Cw https://youtu.be/sjBR-YL828o

Types of software and the different methodologies used to develop software		Content clarification	Links to other topics
1.2.3 Introduction to programming	a) Procedural programming language techniques: • program flow • variables and constants • procedures and functions • arithmetic, Boolean and assignment operators • string handling • file handling. b) Assembly language (including following and writing simple programs with Little Man Computer). See appendix 5d	Candidates need to have knowledge and experience of using a procedural programming language for example Python, VB.NET etc. There is no substitute for practical experience when learning the content for this section. Candidates need to understand how to control the flow of a program (sequence, iteration and selection). Candidates need to understand 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. Candidates need to understand the benefits of using constants over variables. Candidates need to understand the role of sub-programs (procedures and functions) in a program, how these can be used to reduce the amount of code and improved the efficiency. Candidates need to understand the differences between procedures and functions, and be able to read, write and trace programs using both procedures and functions. Candidates need to have experience of using a range of arithmetic (+, -, /, *, MOD, DIV) operators, Boolean (AND, OR, NOT, ==, >, <, =, >=, <=, !=) operators and assignment operator (=). Candidates need to be able to read, trace and write programs using these 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. Candidates need to have experience of using a range of string handling functions and 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.) Candidates need to have experience of 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. Candidates need to have an understanding of the purpose and need for assembly language. Candidates need to be familiar with the instructions given in Appendix 5d. Candidates should be able to read, write, trace and amend programs written in the Little Man Computer language.	**1.1.1 Structure and function of the processor** **1.4.2 Data Structures** **2.2.1 Programming techniques** **2.3.1 Algorithms** https://www.techopedia.com/definition/8982/procedural-language https://youtu.be/oKUTv944SXc http://peterhigginson.co.uk/lmc/ http://www.yorku.ca/sychen/research/LMC/

1.3. Exchanging data

How data is exchanged between different systems		Content clarification	Links to other topics
1.3.1 Databases	a) Relational database, flat file, primary key, foreign key, secondary key, entity relationship modelling See appendix 5e b) Methods for capturing, selecting, managing and exchanging data.	Candidates need to understand what is meant by a database. Candidates should be familiar with basic database terminology such as fields, records and tables. Candidates should know the difference between a flat file and a relational database, and be able to explain the benefits and limitations of each approach. Candidates should have experience of setting up and using both a flat file, and relational database. Candidates should know what is meant by a primary key, foreign key and secondary key and how each are used in a database. Candidates should be able produce and follow Entity Relationship (ER) diagrams which include 1:1, 1:M and M:M relationships. Candidates should be able to identify how tables should be linked. Candidates need to have an awareness of 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.	**2.1.2 Thinking ahead** https://www.lucidchart.com/pages/er-diagrams https://youtu.be/ob7Zy8NgVK4
1.3.2 Networks	a) Characteristics of networks and the importance of protocols and standards b) Internet structure: The TCP/IP Stack. DNS Protocol layering. LANs and WANs. Packet and circuit switching. c) Client-server and Peer to peer.	Candidates need to understand the definition and purpose of a network. Candidates need to understand the purpose of, and importance of using, protocols. Candidates should be able to discuss examples of protocols that may be used in a network/the internet (but will not be asked to recall information about any specific protocol). Candidates should understand the term standard, and the purpose and need for standards in a network (or any situation where data is transferred). Candidates need to understand 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. Candidates need to understand how data is transmitted on the Internet, the use of IP addresses and packets in the transfer of data. (NB: Candidates are not expected to be familiar with the OSI model). Candidates are expected to understand the terms LAN and WAN. Candidates need to understand how the Domain Name System is used to find the IP address of a URL. Candidates need to understand the purpose, function, benefits and drawbacks of both packet and circuit switching. Candidates need to understand the difference between a client-server and peer-to-peer network. Candidates need to know the benefits and drawbacks of each type of network and be able to recommend one for a given scenario.	**1.3.3 Web Technologies** **1.5.2 Ethical Issues** https://youtu.be/1gdrZwBouOs http://www.computerworld.com/article/2593382/networking/networking-packet-switched-vs-circuit-switched-networks.html

How data is exchanged between different systems		Content clarification	Links to other topics
1.3.3 Web Technologies	a) HTML, CSS and JavaScript. See appendix 5d b) Lossy v lossless compression.	Candidates need to understand the purpose of HTML, CSS and JavaScript. Candidates need to know when each language/markup would be used, and what its purpose and function is. Candidates should have experience of writing webpages using HTML, CSS and JavaScript. Candidates need to be able to recognise the code in Appendix 5d, and be able to read, write, amend and interpret code using HTML, CSS and JavaScript. Candidates need to understand the need for compression (when transferring data over a network). Candidates need to understand the difference between lossy and lossless compression, and the benefits and drawbacks of each type. Candidates need to be able to recommend a type of compression for a given scenario.	**1.3.2 Networks** http://www.w3schools.com/html/ http://www.w3schools.com/js/default.asp http://www.w3schools.com/css/default.asp https://youtu.be/faJFlcVJIDM

1.4. Data types, data structures and algorithms

How data is represented and stored within different structures. Different algorithms that can be applied to these structures		Content clarification	Links to other topics
1.4.1 Data Types	a) Primitive data types, integer, real/floating point, character, string and Boolean b) Represent positive integers in binary. c) Use of Sign and Magnitude and Two's Complement to represent negative numbers in binary d) Addition and subtraction of binary integers. e) Represent positive integers in hexadecimal. f) Convert positive integers between Binary Hexadecimal and denary g) Positive and negative real numbers using normalised floating point representation h) How character sets (ASCII and UNICODE) are used to represent text.	Candidates need to have an understanding of programming data types such as integer, real, Boolean, character, string etc. Candidates need to be able to choose appropriate data types for a situation or given data. Candidates should have experience of programming solutions using these data types. Candidates should have knowledge of how to convert from one data type to another (casting). Candidates should understand 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). Candidates should be able to convert positive whole numbers to binary and from binary to denary. Candidates should know how to store negative numbers using Sign and Magnitude and Two's Complement. Candidates should be able to convert denary numbers to sign and magnitude, and two's complement – and vice-versa. Candidates should be able to perform addition and subtraction on integer binary numbers. (These numbers could be positive or negative using two's complement representation.) Candidates need to have an understanding of 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. Candidates should be able to convert denary numbers to hexadecimal and vice-versa and from binary to hexadecimal and vice-versa. Candidates should have an understanding of how (positive and negative) real numbers are represented in a binary floating-point representation, and should be able to convert 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.) Candidates should understand the need for normalised floating point numbers. Candidates should be able to normalise a floating point number. Candidates should have an understanding of how characters are represented in binary. Candidates should understand the need for a character set and how a computer makes use of a character set. Candidates should be aware of the ASCII and UNICODE character sets and be able to explain the differences between these and the benefits of each. Candidates should be able 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)	**1.2.3 Introduction to programming** http://www.bbc.co.uk/education/guides/zwsbwmn/revision/7 http://thestarman.pcministry.com/asm/hexawhat.html http://www.bbc.co.uk/education/guides/zjfgjxs/revision/5 https://youtu.be/715N3qyrYJk https://youtu.be/YtMv4u-9poQ https://youtu.be/tKZsdbn8XQs

How data is represented and stored within different structures. Different algorithms that can be applied to these structures		Content clarification	Links to other topics
1.4.2 Data Structures	a) Arrays (of up to 3 dimensions), records, lists, tuples b) The properties of stacks and queues.	Candidates should be able to describe what is meant by arrays (up to 3 dimensions), records, lists and tuples. Candidates are expected to be able recognise when they can be used and incorporate them in their programs to store data. Candidates should have an understanding of the purpose and use of a record structure to store data of different data types in a program. Candidates should have experience of using records to store, search, manipulate and retrieve data. Candidates should have an understanding of the purpose and use of a list to store data in a program. Candidates should have experience of using lists to store, search, manipulate and retrieve data. Candidates should have an understanding of the purpose and use of tuples to store data in a program. Candidates should have experience of using tuples to store, search, manipulate and retrieve data. Candidates need to have an understanding of the behaviour of stacks and queues (i.e. LIFO and FIFO).	**1.2.3 Introduction to programming** **2.3.1 Algorithms** https://www.cs.cmu.edu/~adamchik/15-121/lectures/ https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm https://www.tutorialspoint.com/data_structures_algorithms/dsa_queue.htm https://www.cs.bu.edu/teaching/c/stack/array/ https://youtu.be/b8s0-VLkVA0 https://youtu.be/K72XTSusEO0 https://www.youtube.com/watch?v=okr-XE8yTO8 https://www.cs.bu.edu/teaching/c/queue/array/types.html

How data is represented and stored within different structures. Different algorithms that can be applied to these structures		Content clarification	Links to other topics
1.4.3 Boolean Algebra	a) Define problems using Boolean logic. See appendix 5d b) Manipulate Boolean expressions. Including the use of Karnaugh maps to simplify Boolean expressions c) Use logic gate diagrams and truth tables	Candidates should be familiar with AND, OR, NOT and XOR. Candidates should be familiar with the logic of each Boolean operator, and the truth tables. Candidates should be able to construct logic gate diagrams from a Boolean expression and vice-versa. Candidates should be able to construct truth tables from Boolean expressions and logic gate diagrams. Candidates should have an understanding that Boolean expressions can be simplified and should have experience of simplifying expressions using Karnaugh maps. Candidates should be able to create, complete and interpret Karnaugh maps to simplify Boolean expressions.	**1.2.3 Introduction to programming** **2.2.1 Programming techniques** **2.3.1 Algorithms** http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karnaugh.html http://www.32x8.com/ https://youtu.be/uTUV-VnKAdM https://youtu.be/hflkW-D3hrM https://youtu.be/osI68WZz_CM https://www.youtube.com/watch?v=hflkW-D3hrM https://youtu.be/7SCp9Flmz4Y

1.5. Legal and ethical issues

The individual (moral) and social (ethical) opportunities and risks of digital technology and the laws surrounding the use of computers and ethical, moral and cultural issues that can or may in the future arise from the use of computers		Content clarification	Links to other topics
1.5.1 Computing related laws	a) Data Protection Act 1998. b) Computer Misuse Act 1990. c) Copyright and Patents Act 1988. d) Regulation of Investigatory Powers Act 2000.	Candidates need to have an understanding of 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. Candidates need to understand the purpose and principles of the Computer Misuse Act, including the actions that it prohibits. Candidates need to understand the purpose and principles of the Copyright and Patents Act, including the actions that it prohibits. Candidates need to understand the purpose and principles of the Regulation of Investigatory Powers Act, and what this allows in interception and monitoring of electronic communication. Candidates need to understand how the regulations impact organisations and the use of computers and electronic communication. We are aware the law is constantly changing and some of the mentioned laws/acts (most notably the DPA) are likely to change over the course of the specification. Answers will be accepted that use an interpretation of the law based on when the specification was started or when the examination was sat. Please note, a question that requires an extended response can be asked from any area within the specification. These questions are assessed using a level of response framework, where the response requires specific areas to have been covered to allow it to reach that level. In this area of the specification, for example, a question may be asked on the social and ethical impacts of a specific technology in a specific scenario or context. To gain the highest level, candidates would need to discuss whichever moral or social elements are relevant in the question, and because there is a context, every point they make should be in the context given, or related after to the context. If the question requires a judgment, or conclusion, then this needs to be given and justified against the context given.	http://www.legislation.gov.uk/ukpga/2000/23/contents https://youtu.be/R1ymVnk5XZA https://youtu.be/DdZWxIlYKQk

The individual (moral) and social (ethical) opportunities and risks of digital technology and the laws surrounding the use of computers and ethical, moral and cultural issues that can or may in the future arise from the use of computers		Content clarification	Links to other topics
1.5.2 Ethical Issues	The individual moral, social, ethical and cultural opportunities and risks of digital technology: • Computers in the workforce • Automated decision making • Artificial intelligence • Environmental effects • Censorship and the Internet • Monitor behaviour • Analyse personal information a) The ability to articulate cultural opportunities and risks of digital technology • Language and differing alphabet and character sets • Use of cultural colour paradigms	In order to prepare for this section we would recommend candidates regularly keep abreast of technological developments in the news. Candidates need to understand what is meant by moral, social, ethical and cultural issues in relation to the use of computers. Candidates need to understand how the use of computers, and the increasing use of computers in the work force has moral, social, ethical and cultural implications and risks to a variety of people such as the employees, employers, society and organisations. Candidates need to understand how the use of computers to make decisions automatically has moral, social, ethical and cultural implications and risks to 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. Candidates need to understand how the development of artificial intelligence has moral, social, ethical and cultural impacts on a variety of people. Candidates need to understand how the environmental effects of computers (such as disposal, energy use) have moral, social, ethical and cultural implications. Candidates need to understand how the Internet and censorship on the Internet has moral, social, ethical and cultural implications. Candidates need to understand the moral, social, ethical and cultural implications of using computers to monitor behaviour (such as CCTV, tracking phone calls, GPS, monitoring emails). Candidates need to understand the moral, social, ethical and cultural implications of using computers to analyse personal information (such as the gathering, storing and analysing of medical records) Candidates need to understand 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. Candidates should understand 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.	**1.3.2 Networks** http://www.bbc.co.uk/ethics/introduction/intro_1.shtml https://aitopics.org/search?filters=taxnodes:Technology%7CInformation%20Technology%7CArtificial%20Intelligence%7CIssues%7CSocial%20%26%20Ethical%20Issues https://philosophynow.org/issues/110/Surveillance_Ethics http://www.informationisbeautiful.net/visualizations/colours-in-cultures/ http://www.empower-yourself-with-color-psychology.com/cultural-color.html

Component 2

2.1. Elements of computational thinking

Understand what is meant by computational thinking		Content clarification	Links to other topics
2.1.1 Thinking abstractly	a) The nature of abstraction. b) The need for abstraction. c) The differences between an abstraction and reality. d) Devise an abstract model for a variety of situations.	Candidates need to understand the term abstraction, it's purpose in the design and creation of computer programs. Candidates need to know about the benefits of abstraction and be able to apply these benefits to specific scenarios. Candidates may be given a scenario and be asked how abstraction can be applied to it, or how it has already been applied. Candidates need an understanding of the differences between reality and abstraction.	**2.3.1 Algorithms** **2.1.2 Thinking ahead** http://www.bbc.co.uk/education/guides/zttrcdm/revision http://whatis.techtarget.com/definition/abstraction
2.1.2 Thinking ahead	a) Identify the inputs and outputs for a given situation. b) Determine the preconditions for devising a solution to a problem. c) The need for reusable program components.	Candidates need to understand that situations require inputs and output, and that outputs can be both digital or in a hard copy format. Candidates may be given a description, diagram, or code for a scenario, and they will need to demonstrate an understanding of what inputs and outputs are needed, and/or are used in that specific scenario. For a description of a program, candidates need to be able to determine what else they need to know before they can produce a solution, for example what information is missing and what else will affect that solution. Candidates need to understand the purpose, benefits and drawbacks of reusable program components. Candidates should understand how these components can be reused, and for a given scenario/program they will need to be able to identify the subprograms that will be needed. Candidates may then be required to write code for these reusable components.	**2.1.1 Thinking abstractly** **2.1.3 Thinking procedurally** **2.1.4 Thinking logically** **2.3.1 Algorithms** https://youtu.be/5dPDrjJFUv8
2.1.3 Thinking procedurally	a) Identify the components of a problem. b) Identify the components of a solution to a problem. c) Determine the order of the steps needed to solve a problem. d) Identify sub-procedures necessary to solve a problem.	Candidates need to be able to deconstruct a program and identify the component parts that will make it up, for example listing the parts or completing a structure diagram. Candidates may be given some component parts and be asked to complete these from a written description or pseudocode for a program. Candidates need to be able to identify the steps that will need to take place to complete the algorithm or program, and be able to write these in a suitable format, or put a given list into the correct order to produce a working program. Candidates may need to write pseudocode or draw a flow chart to show this sequencing of steps. For a given scenario, candidates need to be able to identify where sub-procedures may be used, and then write appropriate pseudocode for these sub-procedures, making use of parameters where appropriate. Candidates may be given a structure diagram that they will need to interpret, or complete to identify these sub-procedures.	**2.1.1 Thinking abstractly** **2.1.2 Thinking ahead** **2.1.4 Thinking logically** **2.3.1 Algorithms** http://www.bbc.co.uk/guides/z8ngr82 https://youtu.be/u7WsR8iTTnI

Understand what is meant by computational thinking		Content clarification	Links to other topics
2.1.4 Thinking logically	a) Identify the points in a solution where a decision has to be taken. b) Determine the logical conditions that affect the outcome of a decision. c) Determine how decisions affect flow through a program.	Candidates need to understand that decisions are made within programs, and they need to be able to identify where these decisions will take place within an algorithm or program, and be able to understand what these decisions are and the impact of these decisions on the algorithm/program and the next (and final) outcomes from the algorithm/program. Candidates need to understand that there can be many different routes through a program, and understand how decisions influence these routes and outcomes.	**2.1.1 Thinking abstractly** **2.1.2 Thinking ahead** **2.1.3 Thinking procedurally** **2.3.1 Algorithms** https://youtu.be/2ybo8KiU32k https://youtu.be/ZrPz-ENUhbs

2.2. Problem solving and programming

How computers can be used to solve problems and programs can be written to solve them *(Learners will benefit from being able to program in a procedural/imperative language.)*		**Content clarification**	**Links to other topics**
2.2.1 Programming techniques	a) Programming constructs: sequence, iteration, branching. b) Global and local variables. c) Modularity, functions and procedures, parameter passing by value and reference. d) Use of an IDE to develop/debug a program.	Candidates need to be able to understand the constructs of sequence, iteration and branching. Candidates must be able to use these constructs independently of each other, and combine them to produce a solution. These include the selection statements of if (include elseif and else) and select case statements. These include both condition based iteration (e.g. while, repeat until) and count controlled iteration (e.g. for) – as well as how condition based can be used as count controlled iteration. Candidates need to be able to read code using these constructs, create code using these constructs and trace code (for example using a trace table). Candidates need to understand the use and need for variables in a program, and must understand the difference, benefits and drawbacks of both global and local variables. Candidates must be able to recognise where local and global variables are used, and the impact that these have on the program, for example the amount of memory used by the program. Candidates need to understand how a program using global variables can be changed to use local variables – and vice-versa. Candidates need to understand what is meant by modular code, and how this can be produced using functions and procedures. Candidates need to understand the differences between functions and procedures and how each is used within a program. Candidates need to be able to read, trace and write code using functions and procedures. Candidates need to understand the purpose and use of parameters within a program, and how they are used in functions and procedures. Candidates will need to be able to read, trace and write code that makes use of parameters. Candidates need to understand the difference between passing a parameter by value and by reference, they need to understand the benefits and drawbacks of each, recommending which should be used for a given situation. Candidates need to be able to read, trace and write code that makes use of parameters passed both by value and by reference. Candidates should have had experience of using an IDE to produce code. Candidates need to understand how an IDE can be used to produce code, and understand the range of features and tools that are within an IDE that can be used to help produce and debug a program.	**2.1.1 Thinking abstractly** **2.1.2 Thinking ahead** **2.1.3 Thinking procedurally** **2.1.4 Thinking logically** **2.3.1 Algorithms** http://vle.moirahouse.co.uk/studentwebsites/ict/theteacherict/newalevel/cp1_2_1.htm http://users.csc.calpoly.edu/~jdalbey/SWE/pdl_std.html https://youtu.be/xBAMBDyDu0s https://youtu.be/tflZNOyF8yA https://youtu.be/1vcCx1ndV2Q

	How computers can be used to solve problems and programs can be written to solve them *(Learners will benefit from being able to program in a procedural/imperative language.)*	**Content clarification**	**Links to other topics**
2.2.2 Software Development	a) Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development. b) The relative merits and drawbacks of different methodologies and when they might be used. c) Writing and following algorithms. d) Different test strategies including black and white box testing and alpha and beta testing e) Test programs that solve problems using suitable test data and end user feedback, justify a test strategy for a given situation.	Candidates need to understand 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). Candidates need to understand the tasks, processes, benefits and drawbacks of each model and the similarities and differences between each. Candidates need to understand where each model is most suitable to use, and be able to justify the use in a situation. Candidates need to be able to write algorithms using flow charts, pseudocode and/or program code. Candidates need to be able to follow the code as shown in the OCR pseudocode guide, but are not expected to write code in this. Candidate's code is not expected to be syntactically correct, but must use appropriate code structures. Candidates should have experience of using black box testing, white box testing, alpha testing and beta testing whilst producing their own programs. Candidates need to understand how each testing strategy can be used in a situation, and the benefits and drawbacks of each method, and apply this to a given situation to recommend appropriate testing strategies. Candidates should have experience of using suitable test data to test their own programs. Candidates need to understand the use of test data and apply this to a given program. Candidates need to understand how dry runs can be used in the development and testing of programs, and be able to use dry runs to test given code. Candidates should understand the need for and importance of end user feedback.	https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm http://agilemethodology.org/ https://www.tutorialspoint.com/sdlc/sdlc_rad_model.htm https://narbit.wordpress.com/2012/06/10/the-differences-between-life-cycle-models-advantages-and-disadvantages/ http://www.webopedia.com/TERM/B/Black_Box_Testing.html http://www.webopedia.com/TERM/W/White_Box_Testing.html

2.3 Algorithms

The use of algorithms to describe problems and standard algorithms	Content clarification	Links to other topics
a) Analysis and design of algorithms for a given situation. b) Standard algorithms (Bubble sort, insertion sort, binary search and linear search). c) Implement bubble sort, insertion sort. d) Implement binary and linear search. e) Representing, adding data to and removing data from queues and stacks. f) Compare the suitability of different algorithms for a given task and data set. g) Analysis and design of algorithms for a given situation. h) Standard algorithms (Bubble sort, insertion sort, binary search and linear search). i) Implement bubble sort, insertion sort. j) Implement binary and linear search. k) Representing, adding data to and removing data from queues and stacks. l) Compare the suitability of different algorithms for a given task and data set.	Candidates need to be able to write algorithms using flow charts, pseudocode and program code. Candidates need to be able to follow the code as shown in the OCR pseudocode guide, but are not expected to write code in this syntax. Candidate's code is not expected to be syntactically correct, but must use appropriate code structures. Candidates need to understand the need for standard sorting algorithms. Candidates need to understand how the sorting algorithms bubble and insertion work and the situations when each can, and cannot be used. Candidates need to be able to use the algorithms to sort data, and complete, write and correct algorithms to perform each sorting algorithm. Candidates need to understand the need for standard searching algorithms. Candidates need to understand how the searching algorithms binary and linear work and the situations when each can, and cannot be used. Candidates need to be able to use the algorithms to search data sets for specific values that may, or may not exist in the data set. Candidates need to understand when each searching algorithm can, and cannot be used. Candidates need to be able to complete, write and correct algorithms to perform each searching algorithm. Candidates should have experience of using the data structures stacks and queues. Candidates need to understand the differences and similarities between stacks and queues. Candidates need to be able to add and remove data from both stacks and queues. Candidates need to understand how pointers are used within stacks and queues. Candidates need to understand how stacks and queues can be implemented in a computer system, for example through the use of an array with pointers. Candidates need to be able to read, correct and write algorithms to add and remove data items, and manipulate data items in a stack and queue. Candidates need to understand how the choice of algorithm can be affected by the data set. Candidates need to understand the impact of specific algorithms on speed and memory use. Candidates are not expected to know about Big O notation, but should be aware of how and when a program can use more memory, or can take longer to run and be able to compare algorithms to determine which will use more/less memory, and which will run faster/slower.	**2.1.3 Thinking procedurally** **2.1.4 Thinking logically** **2.2.1 Programming techniques** https://youtu.be/iPGUYPQWeTI https://youtu.be/uTfiT8Z5tMQ https://youtu.be/NHDuXoaoqEA https://www.cs.cmu.edu/~adamchik/15-121/lectures/Stacks%20and%20Queues/Stacks%20and%20Queues.html https://www.tutorialspoint.com/data_structures_algorithms/stack_algorithm.htm https://www.tutorialspoint.com/data_structures_algorithms/dsa_queue.htm https://www.cs.bu.edu/teaching/c/stack/array/

The small print

We'd like to know your view on the resources we produce. By clicking on the 'Like' or 'Dislike' button you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click 'Send'. Thank you.

Whether you already offer OCR qualifications, are new to OCR, or are considering switching from your current provider/awarding organisation, you can request more information by completing the Expression of Interest form which can be found here: www.ocr.org.uk/expression-of-interest

Looking for a resource?

There is now a quick and easy search tool to help find **free** resources for your qualification:

www.ocr.org.uk/i-want-to/find-resources/

www.ocr.org.uk/alevelreform

OCR Customer Contact Centre

General qualifications

Telephone 01223 553998
Facsimile 01223 552627
Email general.qualifications@ocr.org.uk

OCR is part of Cambridge Assessment, a department of the University of Cambridge. *For staff training purposes and as part of our quality assurance programme your call may be recorded or monitored.*

Cambridge Assessment

LLOYD'S REGISTER · LRQA ISO 9001 UKAS MANAGEMENT SYSTEMS 001