Qualification Accredited

OCR
Oxford Cambridge and RSA

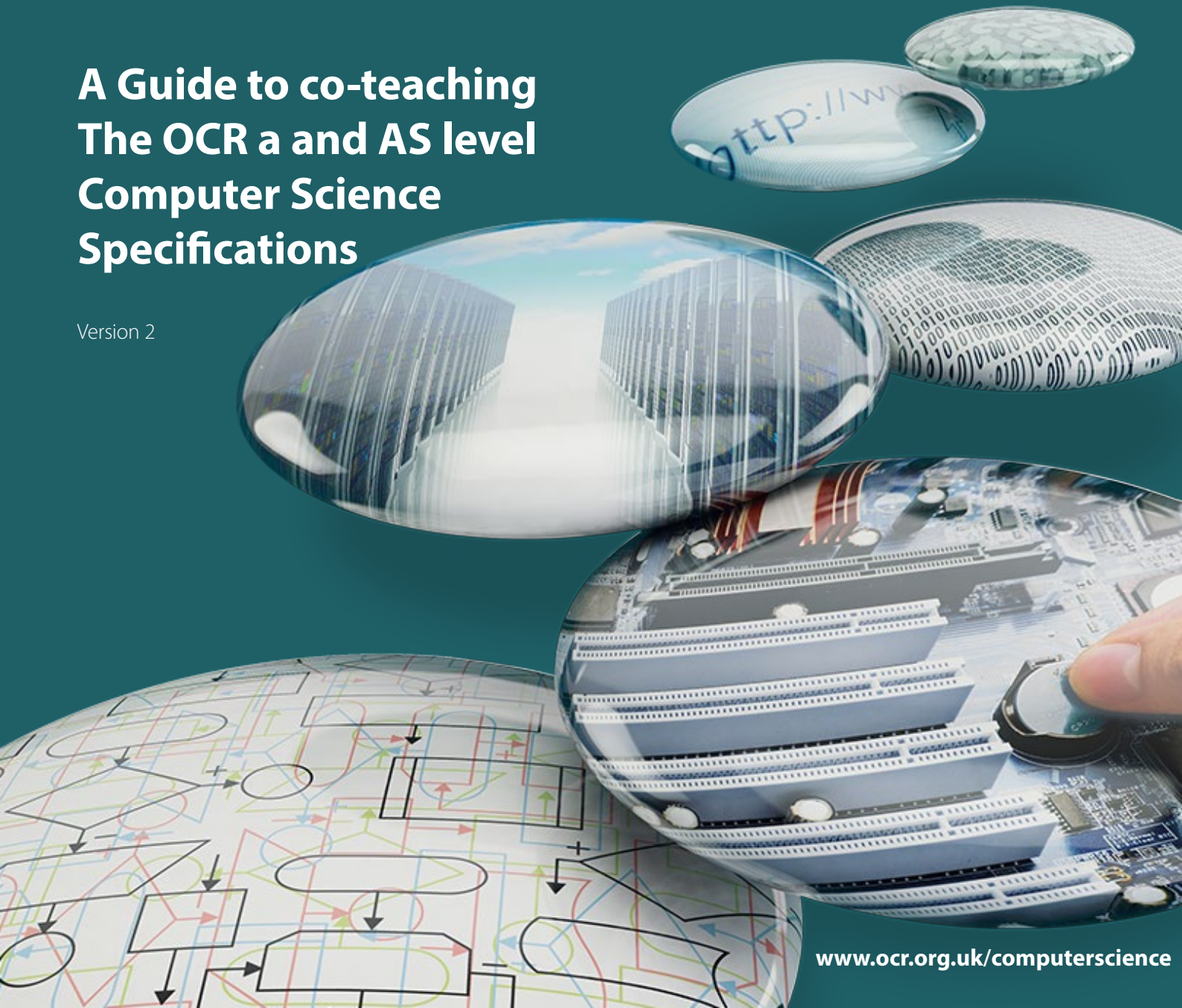
AS and A LEVEL
Co-teach Guide

COMPUTER SCIENCE

H046, H446
For first teaching in 2015

A Guide to co-teaching The OCR a and AS level Computer Science Specifications

Version 2



www.ocr.org.uk/computerscience

CONTENTS

INTRODUCTION

Effective co-teaching of OCR's AS and A Level Computer Science is easily achievable with only minor concessions to the different assessment requirements for each element of the course. The course has been specifically designed to be co-teachable. There is a major overlap between the content explored and skills developed by the two courses, while the assessment methods in the examination papers differ slightly for appropriate AS/A Level demand. There are some areas of teaching eg Object Orientated pseudocode, which are more directly pertinent to A Level students, but this does not impact adversely on the co-teachability of the two courses. In fact both groups of students will benefit throughout from the ways the AS has been designed to build towards the A Level.

All of the components of the AS course have equivalent, albeit more complex, elements in the full A Level: the study of algorithms is based on the same principles as for A Level. In the example curriculum plan it is suggested the programming is studied first and used to develop the underpinning knowledge, skills and concepts for a computer science course.

Week Number	A Level	AS Level
1	1.1.3 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.	1.1.3 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.
2	2.1.2 a) Identify the inputs and outputs for a given situation. b) Determine the preconditions for devising a solution to a problem. c) The nature, benefits and drawbacks of caching. d) The need for reusable program components.	2.1.2 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. 2.2.1 a) Programming constructs: sequence, iteration, branching. b) Global and local variables. c) Modularity, functions and procedures, parameter passing by value and by reference. d) Use of an IDE to develop/debug a program.
3	2.2.1 a) Programming constructs: sequence, iteration, branching. b) Recursion, how it can be used and compares to an iterative approach. c) Global and local variables. d) Modularity, functions and procedures, parameter passing by value and by reference. e) Use of an IDE to develop/debug a program.	

Week Number	A Level	AS Level
4	1.2.3 a) Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development. b) Writing and following algorithms. c) The relative merits and drawbacks of different methodologies and when they might be used. 2.1.4 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.	1.2.3 a) Procedural programming language techniques: • program flow • variables and constants • procedures and functions • arithmetic, Boolean and assignment operators • string handling • file handling. b) Little Man Computer (including following and writing simple programs with Little Man Computer). 2.1.4 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.
5 **6**	1.1.1 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): How this relates to assembly language programs. b) The Fetch-Decode-Execute Cycle. c) The use of pipelining in a processor to improve efficiency. 2.1.4 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. 1.1.2 a) The differences between and uses of CISC and RISC processors. b) GPUs and their uses (including those not related to graphics). c) Multicore and Parallel systems.	1.1.1 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): How this relates to assembly language programs. b) The Fetch-Decode-Execute Cycle. 2.1.4 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. 1.1.2 a) The differences between and uses of CISC and RISC processors. b) Multicore and Parallel systems.

Week Number	A Level	AS Level
7	1.2.1 a) The function and purpose of operating systems. b) Memory Management (paging, segmentation and virtual memory). c) Interrupts. d) Scheduling: Round Robin, First come first served, Multi-level feedback queues, shortest job first and shortest remaining time.	1.2.1 a) The function and purpose of operating systems. b) Memory Management (paging, segmentation and virtual memory). c) Interrupts. d) Scheduling: Round Robin, First come first served, Multi-level feedback queues, shortest job first and shortest remaining time.
8	e) Distributed, Embedded, Multi-Tasking, Multi-User and Real Time operating systems. f) BIOS. g) Device drivers. h) Virtual machines. 2.2.1	e) Distributed, Embedded, Multi-Tasking, Multi-User and Real Time operating systems. f) BIOS. g) Device drivers. h) Virtual machines. 2.2.1
9	a) Programming constructs: sequence, iteration, branching. b) Recursion, how it can be used and compares to an iterative approach. c) Global and local variables. d) Modularity, functions and procedures, parameter passing by value and by reference. e) Use of an IDE to develop/debug a program	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.

Week Number	A Level	AS Level
10	1.2.1 a) The function and purpose of operating systems. b) Memory Management (paging, segmentation and virtual memory). c) Interrupts. 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. 1.1.2 a) The differences between and uses of CISC and RISC processors. b) GPUs and their uses (including those not related to graphics). c) Multicore and Parallel systems.	1.2.1 a) The function and purpose of operating systems. b) Memory Management (paging, segmentation and virtual memory). c) Interrupts. 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. 1.1.2 a) The differences between and uses of CISC and RISC processors. b) Multicore and Parallel systems.

Week Number	A Level	AS Level
11	1.2.4 a) Procedural languages. b) Assembly language (including following and writing simple programs with the Little Man Computer instruction set). c) Modes of addressing memory (immediate, direct, indirect and indexed). d) Object-oriented languages (using Java/C++ style pseudocode) with an understanding of classes, objects, methods, attributes, inheritance, encapsulation and polymorphism. 2.2.1 a) Programming constructs: sequence, iteration, branching. b) Recursion, how it can be used and compares to an iterative approach. c) Global and local variables. d) Modularity, functions and procedures, parameter passing by value and by reference. e) Use of an IDE to develop/debug a program.	2.2.1 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.
12 **13** **14** **15**	1.4.1 a) Represent positive integers in binary. b) Use of Sign and Magnitude and Two's Complement to represent negative numbers in binary. c) Addition and subtraction of binary integers. d) Represent positive integers in hexadecimal. e) Representation and normalisation of floating point numbers in binary. f) Floating point arithmetic, positive and negative numbers, addition and subtraction. g) Bitwise manipulation and masks: shifts, combining with AND, OR, and XOR. h) How character sets (ASCII and UNICODE) are used to represent text. 2.1.3 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.	1.4.1 a) Represent positive integers in binary. b) Addition and subtraction of binary integers. c) Represent positive integers in hexadecimal. d) How character sets (ASCII and UNICODE) are used to represent text. 2.1.3 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.

Week Number	A Level	AS Level
16	1.4.2 a) Arrays (of up to 3 dimensions). b) The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table. c) How to create, traverse, add data to and remove data from the data structures mentioned above. (NB this can be **either** using arrays and procedural programming **or** an object-oriented approach). 1.5.1 a) Data Protection Act. b) Computer Misuse Act. c) Copyright and Patents Act. d) Regulation of Investigatory Powers Act.	1.4.2 a) Arrays (of up to 3 dimensions). b) The properties of stacks and queues. 1.5.1 a) Data Protection Act. b) Computer Misuse Act. c) Copyright and Patents Act. d) Regulation of Investigatory Powers Act. 1.5.2 These include but are not limited to: a) Computers in the workforce b) Automated decision making c) Artificial intelligence d) Environmental effects e) Censorship and the Internet.
17	1.5.2 a) Arrays (of up to 3 dimensions). b) The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table. c) How to create, traverse, add data to and remove data from the data structures mentioned above. (NB this can be **either** using arrays and procedural programming **or** an object-oriented approach).	

Week Number	A Level	AS Level
18	1.4.2 a) Arrays (of up to 3 dimensions). b) The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table. c) How to create, traverse, add data to and remove data from the data structures mentioned above. (NB this can be either using arrays and procedural programming or an object-oriented approach). 1.2.3 a) Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development. b) Writing and following algorithms. c) The relative merits and drawbacks of different methodologies and when they might be used. 2.1.1 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.	1.4.2 a) Arrays (of up to 3 dimensions). b) The properties of stacks and queues. 1.2.3 a) Procedural programming language techniques: • program flow • variables and constants • procedures and functions • arithmetic, Boolean and assignment operators • string handling • file handling. b) Little Man Computer (including following and writing simple programs with Little Man Computer). 2.1.1 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.
19		
20		

Week Number	A Level	AS Level
21	1.4.3 a) Define problems using Boolean logic. b) Use the following rules to derive or simplify statements in Boolean algebra: De Morgan's Laws, distribution, association, commutation, double negation. 2.2.2 a) Features that make a problem solvable by computational methods. b) Problem Recognition. c) Problem Decomposition. d) Use of divide and conquer. e) Use of abstraction. f) Learners should apply their knowledge of: • backtracking • data mining • heuristics • performance modelling • pipelining • visualisation to solve problems.	1.4.3 a) Define problems using Boolean logic. 2.2.2 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. 2.3.1 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.
22		
23	2.3.1 a) Analysis and design of algorithms for a given situation. b) The suitability of different algorithms for a given task and data set, in terms of execution time and space. c) Comparison of the complexity of algorithms. d) Algorithms for the main data structures, (Stacks, queues, trees, linked lists, depth-first (post-order) and breadth-first traversal of trees). e) Standard algorithms (Bubble sort, insertion sort, merge sort, quick sort, Dijkstra's shortest path algorithm, A* algorithm, binary search and linear search).	

Week Number	A Level	AS Level
24	1.4.3 a) Define problems using Boolean logic. b) Use the following rules to derive or simplify statements in Boolean algebra: De Morgan's Laws, distribution, association, commutation, double negation. 2.3.1 a) Analysis and design of algorithms for a given situation. b) The suitability of different algorithms for a given task and data set, in terms of execution time and space. c) Comparison of the complexity of algorithms. d) Algorithms for the main data structures, (Stacks, queues, trees, linked lists, depth-first (post-order) and breadth-first traversal of trees). e) Standard algorithms (Bubble sort, insertion sort, merge sort, quick sort, Dijkstra's shortest path algorithm, A* algorithm, binary search and linear search).	1.4.3 a) Define problems using Boolean logic. 2.3.1 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.
25		
26	1.3.1 a) Lossy vs Lossless compression. b) Run Length Encoding and dictionary coding for lossless compression. c) Symmetric and asymmetric encryption. d) Different uses of hashing. 1.3.2 a) Relational database, flat file, primary key, foreign key, secondary key, normalisation and indexing. b) Methods of capturing, selecting, managing and exchanging data. c) Normalisation to 3NF. d) SQL - Interpret and modify (list of key words). e) Referential Integrity. f) Transaction processing, ACID (Atomicity, Consistency, Isolation, Durability), record locking and redundancy.	1.3.1 a) Relational database, flat file, primary key, foreign key, secondary key. b) Methods for capturing, selecting, managing and exchanging data.
27		

Week Number	A Level	AS Level
28	1.3.3 a) The TCP/IP Stack. b) Protocol layering. c) LANs and WANs. d) Packet and circuit switching. e) Protocols. f) Client-server and Peer to peer.	1.3.1 a) Relational database, flat file, primary key, foreign key, secondary key. b) Methods for capturing, selecting, managing and exchanging data. 1.3.2 a) The TCP/IP Stack. b) Protocol layering. c) LANs and WANs. d) Packet and circuit switching. e) Protocols. f) Client-server and Peer to peer.
29	1.3.2 a) Relational database, flat file, primary key, foreign key, secondary key, normalisation and indexing. b) Methods of capturing, selecting, managing and exchanging data. c) Normalisation to 3NF. d) SQL - Interpret and modify (list of key words). e) Referential Integrity. f) Transaction processing, ACID (Atomicity, Consistency, Isolation, Durability), record locking and redundancy.	
30	1.3.4 a) HTML, CSS and JavaScript. b) Search engine indexing. c) PageRank Algorithm. d) Server and client side processing.	1.3.3 a) HTML, CSS and JavaScript. b) Lossy v lossless compression.
31		
32		

Week Number	A Level	AS Level
33	1.2.2 a) The nature of applications. b) Utilities. c) Open source vs Closed source. d) Translators: Interpreters, compilers and assemblers. e) Stages of compilation (Lexical Analysis, Syntax Analysis, Code Generation and Optimisation). f) Linkers and loaders.	1.2.2 a) The nature of applications. b) Utilities. c) Open source vs Closed source. d) Translators: Interpreters, compilers and assemblers.
34	**Revise for exams**	
35		

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.

If you do not currently offer this OCR qualification but would like to do so, please complete the Expression of Interest Form which can be found here: www.ocr.org.uk/expression-of-interest

OCR Resources: *the small print*
OCR's resources are provided to support the teaching of OCR specifications, but in no way constitute an endorsed teaching method that is required by the Board and the decision to use them lies with the individual teacher. Whilst every effort is made to ensure the accuracy of the content, OCR cannot be held responsible for any errors or omissions within these resources. We update our resources on a regular basis, so please check the OCR website to ensure you have the most up to date version.

OCR acknowledges the use of the following content:
Square down and Square up: alexwhite/Shutterstock.com

Please get in touch if you want to discuss the accessibility of resources we offer to support delivery of our qualifications: resources.feedback@ocr.org.uk

We will inform centres about any changes to the specification. We will also publish changes on our website. The latest version of our specification will always be the one on our website (www.ocr.org.uk) and this may differ from printed versions.

ocr.org.uk/alevelreform
OCR customer contact centre

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