GCE AS/A LEVEL

wjec cbac

WJEC GCE AS/A LEVEL in COMPUTER SCIENCE

ACCREDITED BY WELSH GOVERNMENT

SPECIFICATION

Teaching from 2015

For award from 2016 (AS)
For award from 2017 (A level)

Version 2 March 2019

This Welsh Government regulated qualification is not available to centres in England.

SUMMARY OF AMENDMENTS

Version	Description	Page number
2	'Making entries' section has been amended to clarify resit rules and the carry forward of NEA marks.	34

WJEC GCE AS and A Level in COMPUTER SCIENCE

For teaching from 2015
For AS award from 2016
For A level award from 2017

This specification meets the GCE AS and A Level Qualification Principles which set out the requirements for all new or revised GCE specifications developed to be taught in Wales from September 2015.

		Page
Summary of assessment		2
1.	Introduction	4
1.1	Aims and objectives	4
1.2	Prior learning and progression	5
1.3	Equality and fair assessment	5
1.4	Welsh Baccalaureate	6
1.5	Welsh perspective	6
2.	Subject content	7
2.1	AS units	7
2.2	A2 units	16
3.	Assessment	28
3.1	Assessment objectives and weightings	28
3.2	Arrangements for Unit 2 on-screen examination	29
3.3	Arrangements for Unit 5 non-exam assessment	30
4.	Technical information	34
4.1	Making entries	34
4.2	Grading, awarding and reporting	35
Appendices:		
	A: Sample non-exam assessment	36
	B: Assessment grids for non-exam assessment	38
	C: Mathematical skills	48
	D: Conventions followed in specification	49

GCE AS and A LEVEL COMPUTER SCIENCE (Wales)

SUMMARY OF ASSESSMENT

AS (2 units)

AS Unit 1 Fundamentals of Computer Science Written examination: 2 hours 25% of qualification 100 marks
This unit investigates computer architecture, communication, data representation, data structures, software applications, programs, algorithms, logic, programming methodologies and the impact of computer science on society.
AS Unit 2: Practical Programming to Solve Problems On-screen examination: 2 hours 15% of qualification 60 marks
This unit consists of a series of set tasks completed on-screen by candidates. These tasks will assess the practical application of knowledge and understanding and will require the use of Visual Basic.NET, Python or Java as a programming language.

A Level (the above plus a further 3 units)

A2 Unit 3 Programming and System Development Written examination: 2 hours 20% of qualification 100 marks
This unit investigates programs, data structures, algorithms, logic, programming methodologies and the impact of computer science on society.
A2 Unit 4 Computer Architecture, Data, Communication and Applications Written examination: 2 hours 20% of qualification 100 marks
This unit investigates computer architecture, communication, data representation, organisation and structure of data, programs, algorithms and software applications.
A2 Unit 5 Programmed Solution to a Problem Non-exam assessment 20% of qualification 100 marks
Candidates discuss, investigate, design, prototype, refine and implement, test and evaluate a computerised solution to a problem chosen by the candidate which must be solved using original code (programming). This is a substantial piece of work, undertaken over an extended period of time.

This is a unitised specification which allows for an element of staged assessment. Assessment opportunities will be available in the summer assessment period each year, until the end of the life of the specification.

Unit 1 and Unit 2 will be available in 2016 (and each year thereafter) and the AS qualification will be awarded for the first time in summer 2016.

Unit 3, Unit 4 and Unit 5 will be available in 2017 (and each year thereafter) and the A level qualification will be awarded for the first time in summer 2017.

Qualification Number listed on The Register:
GCE AS: 601/5391/X
GCE A level: 601/5345/3

Qualifications Wales Approval Number listed on QiW:
GCE AS: C00/0723/2
GCE A level: C00/0722/5

GCE AS and A LEVEL COMPUTER SCIENCE

1 INTRODUCTION

1.1 Aims and objectives

The WJEC AS and A Level in Computer Science encourages learners to develop:

- an understanding of, and the ability to apply, the fundamental principles and concepts of computer science, including abstraction, decomposition, logic, algorithms and data representation
- the ability to analyse problems in computational terms through practical experience of solving such problems, including writing programs to do so
- the capacity for thinking creatively, innovatively, analytically, logically and critically
- the capacity to see relationships between different aspects of computer science
- mathematical skills – see Appendix C
- the ability to articulate the individual (moral), social (ethical), legal and cultural opportunities and risks of digital technology.

Computers are widely used in all aspects of business, industry, government, education, leisure and the home. In this increasingly technological age, a study of computer science, and particularly how computers are used in the solution of a variety of problems, is not only valuable to the learners but also essential to the future well-being of the country.

Computer science integrates well with subjects across the curriculum. It demands both logical discipline and imaginative creativity in the selection and design of algorithms and the writing, testing and debugging of programs; it relies on an understanding of the rules of language at a fundamental level; it encourages an awareness of the management and organisation of computer systems; it extends the learners' horizons beyond the school or college environment in the appreciation of the effects of computer science on society and individuals. For these reasons, computer science is as relevant to a learner studying arts subjects as it is to one studying science subjects.

The WJEC AS and A Level in Computer Science has been designed to give an in-depth understanding of the fundamental concepts of computer science and a broad scope of study opportunities. This specification has been designed to free centres to concentrate on innovative delivery of the course by having a streamlined, uncomplicated, future-proof structure, with realistic technological requirements.

1.2 Prior learning and progression

There are no prior learning requirements. Any requirements set for entry to a course following this specification are at the discretion of centres. It is reasonable to assume that many learners have achieved qualifications equivalent to Level 2 at KS4. Skills in Numeracy/Mathematics, Literacy/English and Information Communication Technology will provide a good basis for progression to this qualification.

Some learners will have already gained knowledge, understanding and skills through their study of Computer Science at GCSE.

Mathematical requirements are specified in the subject criteria and repeated in Appendix C of this specification.

This specification provides a suitable foundation for the study of Computer Science or a related area through a range of higher education courses, progression to the next level of vocational qualifications or employment. In addition, the specification provides a coherent, satisfying and worthwhile course of study for learners who do not progress to further study in this subject.

This specification is not age specific and, as such, provides opportunities for learners to extend their life-long learning.

1.3 Equality and fair assessment

This specification may be followed by any learner, irrespective of gender, ethnic, religious or cultural background. It has been designed to avoid, where possible, features that could, without justification, make it more difficult for a learner to achieve because they have a particular protected characteristic.

The protected characteristics under the Equality Act 2010 are age, disability, gender reassignment, pregnancy and maternity, race, religion or belief, sex and sexual orientation.

The specification has been discussed with groups who represent the interests of a diverse range of learners, and the specification will be kept under review.

Reasonable adjustments are made for certain learners in order to enable them to access the assessments (e.g. application for extra time in a GCE subject where extended writing is required). Information on reasonable adjustments is found in the following document from the Joint Council for Qualifications (JCQ): *Access Arrangements and Reasonable Adjustments: General and Vocational Qualifications.* This document is available on the JCQ website (www.jcq.org.uk).

We will be following the principles set out in this document and, as a consequence of provision for reasonable adjustments, very few learners will have a complete barrier to any part of the assessment.

1.4 Welsh Baccalaureate

In following this specification, learners should be given opportunities, where appropriate, to develop the skills that are being assessed through the Core of the Welsh Baccalaureate:

- Literacy
- Numeracy
- Digital Literacy
- Critical Thinking and Problem Solving
- Planning and Organisation
- Creativity and Innovation
- Personal Effectiveness.

1.5 Welsh perspective

In following this specification, learners should be given opportunities, where appropriate, to consider a Welsh perspective if the opportunity arises naturally from the subject matter and if its inclusion would enrich learners' understanding of the world around them as citizens of Wales as well as the UK, Europe and the world.

2 SUBJECT CONTENT

This specification promotes the integrated study of computer science. It will enable learners to develop a broad range of skills in the areas of programming, system development, computer architecture, data, communication and applications.

The knowledge, understanding and skills are set out in the two columns in the pages that follow. The topic to be studied is in the first column, with the amplification in the second column. There is no hierarchy implied by the order in which content and amplification are presented, nor should the length of the various sections be taken to imply any view of their relative importance.

2.1 AS UNITS

Unit 1: Fundamentals of Computer Science
Written examination: 2 hours
25% of qualification (62.5% of AS qualification)

Unit 2: Practical Programming to Solve Problems
On-screen examination: 2 hours
15% of qualification (37.5% of AS qualification)

1. Hardware and communication

	Identify and describe the hardware and communication elements of contemporary computer systems and how they are connected.
Architecture	Identify and describe the main components of contemporary computer architecture, including Von Neumann architectures.
	Describe different types of memory and caching.
	Describe and explain parallel processing.
Fetch-execute cycle	Describe the fetch-execute cycle showing how data can be read from RAM into registers.
Input / output	Describe the use of contemporary methods and their associated devices for input and output.
	Explain the use of these methods and devices in contemporary computer systems and their suitability in different situations.
Secondary storage	Compare the functional characteristics of contemporary secondary storage devices.
Data storage on disc	Explain fragmentation and its consequences and describe the need for defragmentation.

Networking	Describe networks and how they communicate. Explain the importance of networking standards. Describe the importance and the use of a range of contemporary protocols including HTTP, FTP, SMTP, TCP/IP, IMAP, DHCP, UDP and wireless communication protocols. Explain the role of handshaking.
Internet	Describe the internet in terms of a world-wide communications infrastructure.

2. Logical operations

	Draw truth tables for Boolean expressions consisting of AND, OR, NOT and XOR logical operations. Apply logical operations to combinations of conditions in programming, including clearing registers and masking. Simplify Boolean expressions using Boolean identities and rules.

3. Data transmission

	Describe serial and parallel transmission, their advantages and disadvantages. Describe simplex, half duplex and full duplex transmission methods. Explain the need for multiplexing and switching.
Communication networks	Describe, using appropriate network protocols, such as TCP/IP the typical contents of a packet. Explain network collision, network collision detection and how these collisions are dealt with. Describe methods of routing traffic on a network.

4. Data representation and data types

Representation of data as bit patterns	Explain the terms bit, byte and word.

	Describe and use the binary number system and the hexadecimal notation as shorthand for binary number patterns.
Storage of characters	Describe how characters and numbers are stored in binary form. Describe standardised character sets.
Data types	Describe the following different primitive data types, Boolean, character, string, integer and real. Describe the storage requirements for each data type.
Representation of numbers as bit patterns	Apply binary arithmetic techniques. Explain the representation of positive and negative integers in a fixed-length store using both two's complement, and sign and magnitude representation. Describe the nature and uses of floating point form. State the advantages and disadvantages of representing numbers in integer and floating point forms. Convert between real number and floating point form. Describe truncation and rounding, and explain their effect upon accuracy.

5. Data structures

	Describe, interpret and manipulate data structures including arrays (up to two dimensions) and records. Describe the manipulation of records and arrays. Select, identify and justify appropriate data structures for given situations.

6. Organisation of data

	Explain the purpose of files in data processing.
File design	Define a file in terms of records and fields. Explain fixed and variable length fields and records and give examples of the appropriate use of each type.

	Design files and records appropriate for a particular application.
File organisation	Distinguish between master and transaction files.
	Describe sequential, indexed sequential and direct (random) file access.
	Distinguish between the use of serial and sequential file access methods in computer applications.
	Describe and design algorithms and programs for sequential file access and update.
	Explain the need for file security, including file backup, generations of files and transaction logs.
	Describe the need for archiving files.
Data validation and verification	Explain and apply appropriate techniques for data validation and verification.
	Design algorithms and programming routines that validate and verify data.

7. Database systems

	Describe and discuss the benefits and drawbacks of relational database systems and other contemporary database systems.
	Describe the use of primary and foreign keys, indexes and links.
	Explain and apply entity relationship modelling and use it to analyse simple problems.
	Describe the advantages of different users having different views of the data in a database.

8. The operating system

Managing resources	Describe the need for and the role of the operating system kernel in managing resources, including peripherals, processes, memory protection and backing store.
Providing an interface	Describe the need for and the role of the operating system in providing an interface between the user and the hardware.
Managing backing store	Explain the hierarchical structure of a directory and describe file attributes.

Utility software	Explain the need for and use of a range of utility software.
Modes of operation	Describe the main features of batch processing, real time control and real time transaction systems. Identify and describe applications that would be suitable to these modes of operation.
Consideration of human-computer interaction	Explain the need to design systems that are appropriate to the variety of different users at all levels and in different environments.

9. Algorithms and programs

	Explain the term algorithm and describe common methods of defining algorithms, including pseudo-code, flowcharts and structured English.
Variables and constants	Identify and explain the use of constants and variables in algorithms and programs.
Identifiers	Describe why the use of self-documenting identifiers, annotation and program layout are important in programs. Give examples of self-documenting identifiers, annotation and appropriate program layout.
Scope of variables	Describe the scope and lifetime of variables in algorithms and programs.
Parameters	Explain the purpose and effect of procedure calling, parameter passing and return, call by reference and call by value.
Mathematical operations	Identify, explain and use mathematical operations in algorithms, including DIV and MOD.
Sorting	Describe the characteristics of sorting algorithms: bubble sort and insertion sort.
Searching	Explain and apply a linear search algorithm. Explain and apply the binary search algorithm. Describe appropriate circumstances for the use of each searching technique. Follow search and sort algorithms and programs and make alterations to such algorithms. Write search algorithms and programs.
Problem analysis	Analyse a problem using appropriate design approaches.

Programming constructs	Identify, explain and use sequence, selection and repetition in algorithms and programs.
	Identify, explain and use counts and rogue values in algorithms and programs.
	Follow algorithms and programs involving sequence, selection and repetition and make alterations to such algorithms.
	Write algorithms and programs involving sequence, selection and repetition to solve non-standard problems.
Modular programming	Identify and explain the nature, use and possible benefits of standard functions, standard modules and user defined subprograms.
Logical operations in algorithms and programs	Identify, use and explain the logical operators AND, OR, NOT and XOR in algorithms and programs.
Compression	Explain data compression and how data compression algorithms are used.
Testing	Select appropriate test data to dry-run a program or algorithm in order to identify possible errors.
	Explain the purpose of a given algorithm by showing the effects of test data.

10. Principles of programming

	Describe the distinguishing features of different types of programming paradigms, including procedural, event-driven, visual and mark-up languages.
	Describe the role of an object-oriented approach to programming and the relationship between object, class and method.
Levels of computer language	Describe the differences between high-level and low-level languages.
	Identify and describe situations that require the use of a high-level or a low-level language.
Types of computer language	Identify and justify which type of language would be best suited to develop a solution to a given problem.

11. Systems analysis

Feasibility	Describe the purpose of a feasibility study and describe the processes that an analyst would carry out during a feasibility study.

	Explain that proposed solutions must be cost effective, developed to an agreed time scale and within an agreed budget.
Investigation and Analysis	Describe the different methods of investigation including direct observation, questionnaires, study of existing documentation and interview.
	Analyse of a problem using appropriate techniques of abstraction and decomposition.
Design	Represent and interpret systems in an appropriate diagrammatic form showing the flow of data and the information processing requirements.
	Describe the selection of suitable software and hardware to address the requirements of a problem.
Changeover	Describe the various methods of changeover: direct, pilot, phased and parallel, identify the most suitable in a given situation and their relative merits.
Program testing	Describe the use of alpha, beta and acceptance testing.
Maintenance	Describe the nature and use made of perfective, adaptive and corrective maintenance.
Backup and recovery	Describe different procedures for backing up and recovering data.
Documentation	Describe the contents and use made of user documentation and maintenance documentation, including annotated listings, variable lists, algorithms and data dictionaries.

12. Software engineering

Software tools	Explain the role of Integrated Development Environment (IDE) tools in developing and debugging programs.

13. Program construction

Compilers, interpreters and assemblers	Describe the principal stages involved in the compilation process: lexical analysis, symbol table construction, syntax analysis, semantic analysis, code generation and optimisation.

14. The need for different types of software systems and their attributes

Types of software	Explain the use of a range of types of software, including open source software, bespoke and off-the-shelf.

Industrial, technical and scientific	Describe the role of the computer in weather forecasting, computer aided design, robotics and the use of computer generated graphics and animation.
Expert systems	Explain the purpose, use and significance of expert systems.

15. Practical programming

Designing programs	Design all the required data structures to a given problem and fully define them in terms of fieldnames, data types, key fields and requirements for validation. Consider and fully describe the methods of access. Design programming routines to be used to handle and process data for a given problem. Document these designs using a structured convention such as pseudo-code.
Develop programs	Use a documented design to produce a functional prototype to a given problem. Demonstrate understanding of code used by producing annotated listings.
Evaluate programs	Identify the successful features of a system and make specific suggestions for improving less successful areas of the system.

16. Data security and integrity processes

Privacy and security	Describe the dangers that can arise from the use of computers to manage files of personal data. Describe contemporary processes that protect the security and integrity of data including standard clerical procedures, levels of permitted access, passwords for access and write-protect mechanisms.
Disaster planning	Describe the various potential threats to computer systems. Describe contingency planning to recover from disasters.
Malicious and accidental damage	Describe malicious and accidental damage to data and identify situations where either could occur.

17. Economic, moral, legal, ethical and cultural issues relating to computer science

	Describe social and economic changes occurring as a result of developments in computing and computer use, and their moral, ethical, legal, cultural and other consequences.
Effect on employment	Describe the possible effects of computers on the nature of employment in the computing industry and wider society.
Legislation	Explain how relevant legislation impacts on security, privacy, data protection and freedom of information.

2.2 A2 UNITS

Unit 3: Programming and System Development
Written examination: 2 hours
20% of qualification

1. Data structures

Describe, interpret and manipulate data structures including arrays (up to three dimensions), stacks, queues, trees, linked lists and hash tables.

Describe the manipulation of arrays (up to three dimensions).

Represent the operation of stacks and queues using pointers and arrays.

Represent the operation of linked lists and trees using pointers and arrays.

2. Logical operations

Draw truth tables for Boolean expressions, including NAND and NOR logical operations.

Apply logical operations to combinations of conditions in programming, including clearing registers, masking and encryption.

Simplify Boolean expressions using Boolean identities, rules and De Morgan's laws.

3. Algorithms and programs

Explain the term algorithm and describe common methods of defining algorithms, including pseudo-code and flowcharts.

Recursion	Explain the use of recursion in algorithms and programs and consider the potential elegance of this approach.
Validation and verification	Identify, explain and apply appropriate techniques of validation and verification in algorithms and programs.
Sorting	Explain the need for a variety of sorting algorithms both recursive and non-recursive.

	Describe the characteristics of sorting algorithms, including quicksort.
	Explain the effect of storage space required, number of comparisons of data items, number of exchanges needed and number of passes through the data on the efficiency of a sorting algorithm.
	Use Big O notation to determine the efficiency of different sorting algorithms in terms of their time and space requirements and to compare the efficiency of different sorting algorithms.
Searching	Explain and apply a shortest-path algorithm.
	Use Big O notation to determine the efficiency of linear and binary searches in terms of execution time and space requirements and to compare the efficiency of different searching algorithms.
	Follow search and sort algorithms and programs and make alterations to such algorithms.
	Write search and sort algorithms and programs.
Logical operations in algorithms and programs	Identify, use and explain logical operators in algorithms and programs, including NAND and NOR.
Traversal of data structures	Write and interpret algorithms used in the traversal of data structures.
Compression	Explain data compression and how data compression algorithms are used.
	Compare and explain the efficiency of data compression algorithms in terms of compression ratio, compression time, decompression time and saving percentage.
Testing	Select appropriate test data.
	Dry-run a program or algorithm in order to identify possible errors, including logical errors.
	Explain the purpose of a given algorithm by showing the effects of test data.
Comparing algorithms	Use Big O notation to determine the complexity and efficiency of given algorithms in terms of their execution time, their memory requirements and between algorithms that perform the same task.

4. Principles of programming

Explain the nature and relative advantages of different programming paradigms, and identify possible situations where they may be used.

Describe the role of an object-oriented approach to programming and the relationship between object, class and method.

Describe the need for the standardisation of computer languages, and the potential difficulties involved in agreeing and implementing standards.

Identify ambiguities in natural language and explain the need for computer languages to have an unambiguous syntax.

Interpret and use formal methods of expressing language syntax: syntax diagrams and Backus-Naur form (extended Backus-Naur form is not to be used).

5. Systems analysis

Approaches	Describe different appropriate approaches to analysis and design, including Waterfall and Agile.
Documentation	Explain at which stage of the development which pieces of documentation would be produced.

6. System design

Human-computer interaction	Discuss contemporary approaches to the problem of communication with computers. Describe the potential for a natural language interface. Describe the problems of ambiguity that can be associated with input that is spoken.
Design validation	Explain the need for a design review to: check the correspondence between a design and its specification; confirm that the most appropriate techniques have been used; confirm that the user interface is appropriate.
Design evaluation	Describe criteria for the evaluation of computer based solutions.

7. Software engineering

Software tools	Describe the types of software tool that have been designed to assist the software engineering process. Explain the role of appropriate software packages in systems analysis, systems specification, systems design and testing. Explain program version management.

8. Program construction

Compilers, interpreters and assemblers	Describe the function of translation programs in making source programs executable by the computer. Describe the purpose and give examples of the use of compilers, interpreters and assemblers, and distinguish between them. Distinguish between and give examples of translation and execution errors.

9. Economic, moral, legal, ethical and cultural issues relating to computer science

Professional behaviour	Describe the role of codes of conduct in promoting professional behaviour.

Unit 4: Computer Architecture, Data, Communication and Applications

Written examination: 2 hours
20% of qualification

1. Hardware and communication

Architecture	Identify and describe the main components of contemporary computer architecture. Describe and explain the limiting factors to parallelisation in parallel processing. Calculate the runtime of given tasks as a result of parallelisation and evaluate the effect of parallelisation.
Assembly language programming	Write simple programs in assembly language and demonstrate how these programs could be executed.
Input / output	Describe and differentiate between voice input for command and control systems to operate a computer system, vocabulary dictation systems for general input and voice print recognition for security. Discuss the suitability of each system in different situations.
Networking	Identify and describe applications where connecting a portable device to a network is required. Describe the hardware required to make a wireless connection and explain how this might be achieved using contemporary wireless technologies.

2. Data transmission

Communication networks	Calculate data transfer rates on a network. Calculate lowest cost routes on a network. Describe the internet in terms of a world-wide communications infrastructure.

3. Data representation and data types

Representation of numbers as bit patterns*	Apply binary arithmetic techniques.*
	Explain the representation of positive and negative integers in a fixed-length store using both two's complement, and sign and magnitude representation.*
	Describe the nature and uses of floating point form.*
	State the advantages and disadvantages of representing numbers in integer and floating point forms.*
	Convert between real number and floating point form.*
	Describe truncation and rounding, and explain their effect upon accuracy.*
	Explain and use shift functions: logical and arithmetic shifts. Interpret and apply shifts in algorithms and programs.
	Describe the causes of overflow and underflow.

4. Organisation and structure of data

File design	Describe how files may be created, organised, updated and processed by programs.
File organisation	Explain the purpose of, and be able to use, a hashing algorithm.
	Compare different hashing algorithms.
	Explain the use of multi-level indexes.
	Explain the techniques used to manage overflow and the need for file re-organisation.

5. Databases and distributed systems

	Explain what is meant by data consistency, data redundancy and data independence.
	Explain what is meant by relational database organisation and data normalisation (first, second and third normal forms).

* Content marked with * also forms part of AS but will be examined at a more demanding level at A2.

	Restructure data into third normal form.
	Explain and apply entity relationship modelling and use it to analyse problems.
	Explain how the data can be manipulated to provide the user with useful information.
Data validation and verification	Explain and apply appropriate techniques for data validation and verification of data in databases.
Searching data	Explain the purpose of query languages.
	Construct and run queries using Structured Query Language (SQL). *See Appendix D for the list of SQL commands and operators with which learners should be familiar.*
Database management systems	Explain the purpose of a database management system (DBMS) and data dictionaries.
Big Data	Explain what is meant by Big Data, predictive analytics, data warehousing and data mining.
Distributed systems	Explain that distribution can apply to both data and processing.
	Describe distributed databases and the advantages of such distribution.

6. The operating system

Types of operating system	Explain the following types of system: batch, single-user (standalone), multi-user (multi-access), multi-tasking and multi-programming.
Interrupts	Describe a range of conditions or events which could generate interrupts.
	Describe interrupt handling and the use of priorities.
	Describe the factors involved in allocating differing priorities.
Memory management and buffering	Explain the reasons for, and possible consequences of, partitioning of main memory.
	Describe methods of data transfer including the use of buffers to allow for differences in speed of devices.

	Describe buffering and explain why double buffering is used.
Scheduling	Describe the principles of high level scheduling: processor allocation, allocation of devices and the significance of job priorities.
	Explain the three basic states of a process: running, ready and blocked.
	Explain the role of time-slicing, polling and threading.

7. The need for different types of software systems and their attributes

Types of software	Explain the use of a range of types of software including safety related, control, expert, wide and local area information exchange systems.
Safety related systems	Explain that some computer applications are safety related and require a high level of dependability, and hence that the development of safety critical systems is a highly specialised field.
Control systems	State the nature and scope of computer control and automation.
	Describe the benefits and implications of automation.
Expert systems	Discuss the possible effects of expert systems on professional groups and the wider community.
Internet and Intranet	Describe the use of search engines on the internet.
	Describe common contemporary applications.
	Discuss the possible effects of the internet upon professional groups and the wider community.

8. Data security and integrity processes

Protecting data integrity	Explain the special security and integrity problems which can arise during online updating of files.
Cryptography	Describe the need for and the purpose of cryptography.
	Describe techniques of cryptography and their role in protecting data.
	Follow algorithms and programs used in cryptography.

	Compare cryptographic methods and their relative strength.
Biometrics	Describe the purpose and use of contemporary biometric technologies. Describe the benefits and drawbacks of biometric technologies. Describe the complexities of capturing, storing and processing biometric data.
Malicious software and mechanisms of attack and defence	Describe types and mechanisms of malicious software and their vectors. Describe black hat hacking, white hat hacking and penetration testing.

Unit 5: Programmed Solution to a Problem
Non-exam assessment
20% of qualification

This unit requires the learners to investigate, design, prototype, implement, test and evaluate a computer solution to a substantial problem of their own choice. The learner's chosen problem must provide sufficient scope for them to access the marks available for each section of the work.

Learners need to investigate their chosen problems in sufficient detail to identify how data is collected, processed and output currently. The current system may be either paper-based or electronic.

Following the identification of their problems, learners should prepare sufficient documentation to allow them to take part effectively in the discussion with their teachers and/or peers.

Notionally this task will require 72 guided learning hours, which includes teaching time.

1. Discussion

Describe, to others, the broad aims of the project specifying possible limitations to the solution of the problem.

Identify and describe, to others, the possible limitations of a solution to the problem.

Consider and use feedback from others to refine understanding of the problem and proposed solution.

2. Investigation

Carry out an investigation of the current system using a variety of appropriate methods.

Research existing solutions to similar problems.

Identify stakeholders of the current system and their requirements for the proposed project.

Analyse data collected for input and processing by the current system.

Identify and describe all outputs from the current system.

Consider the limitations of the current system.

Produce a working specification that summarises the purpose of the project.

Justify the methods to be used in the solution to the problem

Set objectives, including measurable success criteria for the proposed system.

3. Design

Input and output	To achieve each of the stated objectives: Specify, design and document screen layouts, reports and other forms of input and output required to create the user interface.
Data structures and methods of access	Design and document all data structures that will be required to produce the output for the solution to the problem together with the method of accessing the data in that data structure. Ensure that all data entered into the system is valid.
Processing stages	Design programming routines to be used to handle and process data within the proposed solution to achieve each objective. Document these designs using a structured convention such as pseudo-code.

4. Prototype

Justify the areas of the problem to be included in the prototype system.

Produce a range of screens and outputs for the prototype solution.

Create a functioning system that carries out all chosen processes.

Use realistic data for output and storage.

Evaluate the prototype solution.

Make specific suggestions for improvement.

5. Post-prototype refinement of design

Obtain feedback from competent third parties.

Refine designs in light of the evaluation of the prototype solution and feedback received from others.

6. Software development

Refine the prototype using the amended design documentation ensuring that the finished system is functional and suitable for audience and purpose.
Produce annotated listings for the finished system to facilitate future maintenance.

7. Testing

Developmental testing	Produce evidence of testing at each stage of development.
Testing the final system	Produce evidence of all problems encountered and actions taken to overcome these problems. Design a test plan to test: • each individual system function • that each individual function works with typical, extreme or invalid data • the whole system to ensure that the system produces the correct results for the data input.
Actual test runs	Produce annotated test runs that include commentaries on the outcomes of the testing process.

8. Evaluation

Evaluate the system	Produce an evaluation of the programming language used to create the solution. Compare the solution with similar commercially available systems. Identify the successful features of the system and make specific suggestions for improving less successful areas of the system. Describe the strengths and weaknesses of own performance in the design and prototyping of the solution. Describe changes of approach that would be adopted to solve a similar problem.

3 ASSESSMENT

3.1 Assessment objectives and weightings

Below are the assessment objectives for this specification. Learners must:

AO1
Demonstrate knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation.

AO2
Apply knowledge and understanding of the principles and concepts of computer science, including to analyse problems in computational terms.

AO3
Design, program and evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions.

Assessment objective weightings are shown below as a percentage of the full A level, with AS weightings in brackets.

Unit	Unit Weighting	AO1	AO2	AO3
AS Unit 1	25% (62.5%)	15.0% (37.5%)	8.0% (20%)	2.0% (5%)
AS Unit 2	15% (37.5%)	-	9.0% (22.5%)	6.0% (15%)
A2 Unit 3	20%	10.0%	7.5%	2.5%
A2 Unit 4	20%	10.0%	7.5%	2.5%
A2 Unit 5	20%	-	3.0%	17.0%
Total	**100%**	**35.0%**	**35.0%**	**30.0%**

3.2 Arrangements for Unit 2 on-screen examination

This assessment will be carried out in accordance with the instructions set out in 'Instructions for conducting on-screen tests', Appendix 1 of *General, Vocational and Diploma Qualifications: Instructions for conducting examinations* (Joint Council for Qualifications). This document is available on the JCQ website (www.jcq.org.uk).

Programming languages for Unit 2

WJEC will support the following programming languages:

- Visual Basic.NET
- Python
- Java

Some tasks in Unit 2 will require work to be completed using Visual Basic.NET, Python or Java which have integrated development environments (IDE) freely available for legal download.

Centres will be required to inform WJEC of their choice of language and IDE at the start of the course.

WJEC will supply a paper copy of the assessment tasks and a file/files for each candidate.

Candidates will need access to a computer with:

- A 'clean' user area or storage device on which to save their work
- No access to the Internet or email
- Access to a word processor or similar software to produce their responses
- A functional copy of an IDE for their chosen programming language pre-installed
- File(s) supplied by WJEC that have been pre-installed and tested for use in the assessment.

The practical assessment should be carried out under formal supervision, i.e. the candidates must be in direct sight of the supervisor at all times. Use of resources is tightly prescribed and interaction with other candidates is forbidden.

At the end of the assessment candidate work must be copied to a secondary storage medium. Each candidate's work should be saved in a separate folder labelled with the centre number, candidate number and the first two initials of the candidate's surname and the first initial of the candidate's first name. For example Diane Smith (centre number 68999, candidate number 12345) would store her work in a folder named 68999_12345_SM_D.

The secondary storage medium should be sent for marking to an address supplied by WJEC. The centre must also keep an electronic copy of the candidate's work in a secure location in case of loss or damage to the original submission.

3.3 Arrangements for Unit 5 non-exam assessment

Non-exam assessment accounts for 20% of this A Level. In this specification, non-exam assessment is a project. This is a substantial piece of work, undertaken over an extended period of time.

Unit 5 for the A Level examination requires the candidate to discuss, investigate, design, prototype, refine the design, implement, test and evaluate a computerised solution to **a problem chosen by the candidate.**

The work undertaken in this unit will be the discussion, investigation, design, prototyping, refinement of design, implementation, testing and evaluation of a solution to a problem chosen by the candidate. It should demonstrate the discussion, investigation, design, prototyping, refinement of design, implementation, testing and evaluation skills involved in problem solving using a computer and include the development of a piece of work over an extended period of time. The candidate should produce a word-processed report of the work carried out.

During the discussion stage of the project, candidates should discuss their intended task with teachers and peers prior to commencing the investigation; the teacher should note the intended scope of candidates' projects at this stage.

Following the prototype stage of the project, candidates are required to discuss their prototype in technical terms with their teacher and peers. The teacher should make a note of candidate progress and understanding of the work produced. Peers may offer feedback on the work at this stage.

The work for Unit 5 **must** include the development of software in either a general purpose or a special purpose high-level language. The system proposed by the candidate may consist of one integrated program or a suite of related programs.

The candidate's teacher will be expected to mark the candidate's work and note on the *Centre Mark Sheet* the nature of any assistance given and the extent to which the solution actually works as stated in the report. Candidate work and documentation may be annotated by the teacher in order to justify marks awarded.

The assessment grids

When assessing Unit 5, teachers should study the assessment grids (**Appendix B**), which are designed to present a system that links the assessment objectives to marks, and which helps to discriminate clearly between the varying levels of achievement.

The grids will be of most value when used in conjunction with examples of non-exam assessment which will be issued annually to help centres identify the quality of work associated with the various mark bands.

Teachers must make specific reference to the assessment objectives in the comments they write on the work and on the coversheets.

Submission of assessments

Candidate work for Unit 5 should be submitted to WJEC electronically. The submission should include a functioning copy of the solution and supporting documents in portable document format (pdf).

Candidates should make use of the electronic front sheet saved in html format. Each heading on the sheet should be hyperlinked to a pdf version of the write-up for that particular requirement of the specification.

Each candidate's work should be saved in a separate folder labelled with the centre number, candidate number and the first two initials of the candidate's surname and the first initial of the candidate's first name. For example, Diane Smith, Centre number 68999, candidate number 12345 would store her work in a folder named 68999_12345_SM_D.

The candidate may wish to use sub folders to organise their work for each section and must ensure that the front sheet is stored in such a way that the hyperlinks allow the assessor and moderator to access the relevant pdf documents.

The centre assessor should store the candidate's assessment documentation using the same naming convention as above, but with the addition of Unit 5. For example Diane Smith's assessment documentation would be stored as Unit5_68999_12345_SM_D.

Further details will be issued by WJEC each year.

Annotation and supporting evidence

Centres are required to provide information that enables the moderator to check the work against the assessment criteria. This information should be given on the *Centre Mark Sheet.*

Annotation should, therefore:

- describe, in all necessary detail, practical work that is not available, together with comments from the teacher
- explain where candidates have received help beyond the normal learning support that may influence the assessments
- highlight those key areas that have led to the recognition of particular criteria. Reference to the assessment criteria is particularly helpful
- include any other notes that will help the moderator to assess the work.

Supervision and authentication

Unfair practice

Before the course starts, the teacher is responsible for informing candidates of WJEC's regulations concerning malpractice. Candidates must not take part in any unfair practice in the preparation of work required for assessment in the examination. They must understand that to present material copied directly from books or other sources without acknowledgement will be regarded as deliberate deception. Centres must report suspected malpractice to WJEC.

If WJEC is satisfied that the regulations have been breached, the candidate may be disqualified from all subjects. Candidates will be required to certify that they have read and understood the regulations relating to unfair practice.

Supervision of work

Centres must assure WJEC that the assessments submitted are the work of the candidates concerned. For Unit 5 as much work as possible must be undertaken under the direct supervision of teachers.

The teacher responsible for the supervision of the candidate's work must complete a declaration, certifying that the marks submitted were awarded in accordance with the specification and Instructions and Guidance for Teachers, and that she/he is entirely satisfied that the work submitted is that of the candidate concerned.

It is acceptable for parts of a candidate's work to be taken from other sources as long as all such cases are clearly identified in the text and fully acknowledged either on the *Centre Mark Sheet* or in the supporting evidence. Where phrases, sentences or longer passages are quoted directly from a source, candidates should use quotation marks.

Centres entering candidates for A Level Computer Science must accept the obligation to provide sufficient supervision to enable them to give an assurance that every step has been taken to ensure that the work submitted is that of the candidate concerned. When a candidate has need of assistance in completing a particular piece of work, such assistance should be given but the teacher must add appropriate comments on the *Centre Mark Sheet*.

It is expected that the teacher will be involved at the following stages:

- Initial discussion at the time when the unit is being introduced and work is being planned. The final choice of the proposed method of solution should be that of the individual candidate, but the teacher will be expected to discuss the proposed choice so that guidance can be given about suitability and appropriateness before work begins. It is anticipated that such advice will ensure that the solution attempted is neither trivial nor over-ambitious.

- Following the prototype stage, the prototype solution and code produced should be discussed with the candidate. The teacher will be expected to discuss the functionality of the prototype and the candidate's understanding of the code produced. The teacher should note and investigate instances where the candidate is not able to sufficiently explain how the prototype or code in their solution functions. Please see notes on authentication for more information.

- Periodic supervision and discussion as appropriate, e.g. discussion of the availability and use of material, suitable annotations on the work.

- Guidance on the presentation of the unit, including the information to be given on the *Centre Mark Sheet.*

Authentication

It is important that **non-exam assessment is rigorously monitored by centres to ensure that candidates' work is their own**. All candidates are required to sign that the work submitted is their own and teachers/assessors are required to confirm that the work assessed is solely that of the candidate concerned and was conducted under the required conditions. A copy of the authentication form, which forms part of the cover sheet for each candidate's work will be provided by WJEC. It is important to note that **all** candidates are required to sign this form, and not merely those whose work forms part of the sample submitted to the moderator. Malpractice discovered prior to the candidate signing the declaration of authentication need not be reported to WJEC but must be dealt with in accordance with the centre's internal procedures.

Before any work towards the non-exam assessment is undertaken, the attention of candidates should be drawn to the relevant JCQ Notice to Candidates. This is available on the JCQ website (www.jcq.org.uk) and included in *Instructions for Conducting Coursework.* More detailed guidance on the prevention of plagiarism is given in *Plagiarism in Examinations; Guidance for Teachers/Assessors* also available on the JCQ website.

Submission of marks

Centres need to submit marks for internally assessed work online during the summer term of the year when the work is to be submitted for moderation. When the marks have been submitted to WJEC, the online system will apply the sample formula based on the overall rank order for the total entry and immediately identify the sample of learners whose work is selected for moderation.

Internal standardisation and moderation

It is essential that where there is more than one teacher in a centre, work from all teaching groups is standardised internally. This is designed to ensure that the final assessment reflects a single agreed standard for all teaching groups involved. Moderation will normally take place in June. All centres will receive detailed feedback from the moderation.

4 TECHNICAL INFORMATION

4.1 Making entries

This is a unitised specification which allows for an element of staged assessment.

Assessment opportunities will be available in the summer assessment period each year, until the end of the life of the specification.

Unit 1 and Unit 2 will be available in 2016 (and each year thereafter) and the AS qualification will be awarded for the first time in summer 2016.

Unit 3, Unit 4 and Unit 5 will be available in 2017 (and each year thereafter) and the A level qualification will be awarded for the first time in summer 2017.

A qualification may be taken more than once. However, if any unit has been attempted twice and a candidate wishes to enter the unit for the third time, then the candidate will have to re-enter all units and the appropriate cash-in(s). This is referred to as a 'fresh start'. When retaking a qualification (fresh start), a candidate may have up to two attempts at each unit. However, no results from units taken prior to the fresh start can be used in aggregating the new grade(s).

Marks for NEA units may be carried forward for the life of the specification.

If a candidate has been entered for but is absent for a unit, the absence does not count as an attempt. The candidate would, however, qualify as a resit candidate.

The entry codes appear below.

	Title	Entry codes	
		English-medium	Welsh-medium
AS Unit 1	Fundamentals of Computer Science	2500U1	2500N1
AS Unit 2	Practical Programming to Solve Problems	2500U2	2500N2
A2 Unit 3	Programming and System Development	1500U3	1500N3
A2 Unit 4	Computer Architecture, Data, Communication and Applications	1500U4	1500N4
A2 Unit 5	Programmed Solution to a Problem	1500U5	1500N5
AS Qualification cash-in		2500QS	2500CS
A level Qualification cash-in		1500QS	1500CS

The current edition of our *Entry Procedures and Coding Information* gives up-to-date entry procedures.

There is no restriction on entry for this specification with any other WJEC AS or A level specification.

4.2 Grading, awarding and reporting

The overall grades for the GCE AS qualification will be recorded as a grade on a scale A to E. The overall grades for the GCE A level qualification will be recorded as a grade on a scale A* to E. Results not attaining the minimum standard for the award will be reported as U (unclassified). Unit grades will be reported as a lower case letter a to e on results slips but not on certificates.

The Uniform Mark Scale (UMS) is used in unitised specifications as a device for reporting, recording and aggregating candidates' unit assessment outcomes. The UMS is used so that candidates who achieve the same standard will have the same uniform mark, irrespective of when the unit was taken. Individual unit results and the overall subject award will be expressed as a uniform mark on a scale common to all GCE qualifications. An AS GCE has a total of 200 uniform marks and an A level GCE has a total of 500 uniform marks. The maximum uniform mark for any unit depends on that unit's weighting in the specification.

Uniform marks correspond to unit grades as follows:

		Unit grade				
Unit Weightings	**Maximum unit uniform mark**	**a**	**b**	**c**	**d**	**e**
Unit 1 (25%)	125	100	88	75	63	50
Unit 2 (15%)	75	60	53	45	38	30
Unit 3 (20%)	100	80	70	60	50	40
Unit 4 (20%)	100	80	70	60	50	40
Unit 5 (20%)	100	80	70	60	50	40

The uniform marks obtained for each unit are added up and the subject grade is based on this total.

		Qualification grade				
	Maximum uniform marks	**A**	**B**	**C**	**D**	**E**
GCE AS	200	160	140	120	100	80
GCE A level	500	400	350	300	250	200

At A level, Grade A* will be awarded to candidates who have achieved a Grade A (400 uniform marks) in the overall A level qualification and at least 90% of the total uniform marks for the A2 units (270 uniform marks).

APPENDIX A - Sample non-exam assessment

Unit 5 – Programmed solution to a problem
Sample Task

Tasks for Unit 5 will take a variety of forms depending on the scenario chosen by the candidate. The following scenario gives an example of a task that would permit the scope expected from an A level candidate.

Nanthouse Surgery

Nanthouse Surgery is a busy General Practice based in a health centre in a small town. The practice currently has three doctors and two practice nurses, a practice manager and two receptionists.

A large housing development has recently been built near the surgery and there has been a significant increase in the number of patients registered with the practice. The extra workload has forced the doctors and the practice manager to reconsider the way in which the practice functions.

The practice manager is keen to look at time saving solutions such as telephone and online consultations. These consultations would benefit both the doctors and the patients with diagnoses being provided without the patient having to travel to the surgery or the doctor visiting the patient at home.

One of the doctors has an interest in the use of complementary therapies and feels that many of the patients would benefit from such therapies rather than repeated appointments with the doctors. She is considering the introduction of therapies such as acupuncture, nutrition, massage, aromatherapy, counselling and reflexology. Patients will have to pay for the complementary therapy treatments.

The practice has decided to go ahead with both new approaches.

- initially all remote consultations will be online allowing the doctor not only to talk to the patient but to see any issues such as rashes or inflammation.
- only two therapists will be involved in the system at the start, an acupuncturist and a nutritionist. Over time the range of complementary therapies will be expanded.

The practice manager realises that this will impact on the booking system currently in use for patient appointments. It is essential that there is an effective system in place to ensure that all patients receive the care they need.

The practice manager has asked me to develop a system that will include full booking facilities, maintain patient records and keep all sensitive patient data secure. He is particularly concerned that the amount of patient data available to the complementary therapists. The data should be relevant to the treatment being given and that they should not have access to a patient's full medical history.

The practice manager has asked me to create a computer-based solution that will allow the administrative staff to:

- enter and store personal details of each patient as they join the practice
- add more complementary therapists
- book appointments for patients including
    - appointments with a doctor or nurses in the surgery
    - consultations with a doctor online
    - therapy sessions with one of the therapists
- cancel appointments, consultations online and therapy sessions
- send reminder emails or texts to patients on the day before their appointment
- send letters, emails or texts to patients who miss an appointment
- keep all sensitive information safely and securely
- calculate the costs of complementary treatments
- prepare individual invoices for the complementary treatments for each patient

The proposed computer-based solution must allow the doctors and nurses to:

- read patients' medical histories
- enter details of consultations and medicines prescribed
- record any referrals to hospitals or other services
- record any tests that are needed such as blood tests

The proposed computer-based solution must allow the complementary therapists to:

- read only patients' medical records related to the treatment
- record any treatments given to the patient
- record the cost of each treatment session

APPENDIX B -
Assessment grids for non-exam assessment

Banded mark schemes

Banded mark schemes are divided so that each band has a relevant descriptor. The descriptor for the band provides a description of the performance level for that band. Each band contains marks.

Before marking, assessors should first read and annotate a learner's project to pick out the evidence that is being assessed. Once the annotation is complete, the mark scheme can be applied.

This is done as a two stage process.

Stage 1 – Deciding on the band

When deciding on a band, the work should be viewed holistically. Beginning at the lowest band, assessors should look at the appropriate section of the learner's project and check whether it matches the descriptor for that section's mark band. Assessors should look at the descriptor for that band and see if it matches the qualities shown in the learner's work for that section. If the descriptor at the lowest band is satisfied, assessors should move up to the next band and repeat this process for each band until the descriptor matches the work.

If a learner's work covers different aspects of different bands within the mark scheme, a 'best fit' approach should be adopted to decide on the band and then the learner's work should be used to decide on the mark within the band. For instance if work is mainly in band 2 but with a limited amount of band 3 content, the work would be placed in band 2, but the mark awarded would be close to the top of band 2 as a result of the band 3 content. Assessors should not seek to mark candidates down as a result of small omissions in minor areas of their work.

Stage 2 – Deciding on the mark

Once the band has been decided, assessors can then assign a mark. WJEC Eduqas will provide exemplar material already awarded a mark, and this should be used as reference material when assessing the work.

When marking, assessors can use these examples to decide whether a learner's work is of a superior, inferior or comparable standard to the example. Assessors are reminded of the need to revisit the work as they apply the mark scheme in order to confirm that the band and the mark allocated is appropriate to the work submitted.

Where work is not credit worthy, that is, contains nothing of any significance to the project, or has been omitted, no marks should be awarded.

Band	Discussion - AO2.1b
	Max 5 marks
3	**4-5 marks** The candidate has: • identified a substantial problem that provides sufficient scope for the candidate to access the full range of marks • provided a full description of the broad aims of the project using appropriate subject based technical vocabulary • identified the possible limitations of a solution to the problem and is able to describe these in detail using appropriate technical language • fully considered feedback from others and, where appropriate, has used this feedback to refine understanding of the problem and proposed solution.
2	**3 marks** The candidate has: • identified a suitable problem that will provide sufficient scope to produce work at the required level • identified the broad aims of the project • provided a realistic description of these aims • identified the main limitations of a solution to the problem and can describe these to a competent third party • considered feedback from others and has made use of the feedback to increase understanding of the problem and proposed solution.
1	**1-2 marks** The candidate has: • identified a problem that may be limited in scope • identified a limited number of broad aims for the chosen project • provided a description of more than one of the identified broad aims • identified at least one limitation of a solution to the problem • received feedback from others and has made limited use of the advice.
0	**0 marks** Response not credit worthy or not attempted.

Band	Investigation – AO2.1b Max 10 marks
3	**8-10 marks** The candidate has: • made full use of a range of appropriate methods to investigate the existing system • completed a thorough investigation of the current system • carried out extensive desk based research into existing solutions to similar problems and identified the best features and facilities of these solutions to include in their solution to their chosen problem • described the involvement of all stakeholders in the current system and described their requirements for the proposed project • fully analysed data collected for input • analysed in detail the processing carried out by the existing system • given full consideration to all current system outputs • fully considered and explained the limitations of the current system • produced a working specification that clearly summarises the purpose of the project • explained, with technical justification, the methods to be used in the solution • detailed a range of objectives that include success criteria and clearly define the required performance of the proposed system.
2	**4-7 marks** The candidate has: • used several methods to investigate the existing system • completed an investigation of the current system • carried out desk based research into existing solutions to similar problems and identified features to be included in their solution • identified most stakeholders for the project, described them and identified their roles in the current system and their main requirements for the proposed system • analysed the majority of the data that is collected and input into the current system • analysed most of the processing needed to carry out the functions of the existing system • given consideration of the current system in terms of the majority of outputs • described the limitations of the current system • produced a specification that summarises the purpose of the project • described the methods to be used in the solution • put forward objectives that include success criteria.
1	**1-3 marks** The candidate has: • carried out a limited investigation into the current system • given limited consideration to the data that is collected and input into the existing system • carried out desk based research and given limited consideration to existing solutions to similar problems • identified the main stakeholders for the project and described them and their involvement with the current system • given limited consideration to the data that is output by the existing system • identified the main processes carried out by the existing system • given superficial consideration to the limitations of the current system • produced a summary of the purpose of the project • identified a limited number of methods to be used in the solution • put forward limited objectives that give some indication of the scope of the proposed solution.
0	**0 marks** Response not credit worthy or not attempted.

Band	Design – AO3.1a
	Max 15 marks
3	**11-15 marks** The candidate has: • described and justified the breaking down of the problem into sub problems and explained the links between the sub programs and the project objectives • fully identified and described the data to be input to and output from the system • designed in detail the input and output to be produced by the system • given full consideration to the reasons for the data that is to be included in the outputs • fully described all the required files and/or data structures and has fully considered and fully described methods of access • fully described the validation routines to be carried out on the data entered into the system • identified all processes needed to provide a comprehensive solution to the problem • fully explained the relationships between the data and the processes or programs that manipulate and transform the data • fully described the processing routines using an appropriate recognised convention in sufficient detail for implementation by a competent third party. The interconnection between the programs and the files they access has been clearly shown.
2	**6-10 marks** The candidate has: • described how the problem can be broken down into manageable sub problems related to the objectives of the project • identified the majority of the data to be input to and output from the system • designed the majority of the inputs and outputs to be produced by system • described most of the files and/or data structures required to produce the output from the solution • described possible methods of access • described the majority of the validation routines that will be required to ensure all data is valid • identified a sufficient number of processes to provide a working solution to the problem • produced a description of the relationships between the data and the processes or programs that are used to manipulate and transform the data • described the processing routines using a recognised convention. the interconnection between the programs and the files they access has been considered.
1	**1-5 marks** The candidate has: • produced a limited description of the how the problem can be broken down into sub problems • produced a limited description of the data to be input to and output from the system • produced outline designs for the identified input and output facilities • described the main files and/or data structures required to produce a partially functioning solution to the problem • identified several validation rules required to ensure accurate data entry. • identified more than one of the processes that would be included in a working solution to the problem • attempted to describe the routines using a recognised convention.
0	**0 marks** Response not credit worthy or not attempted.

Band	Prototype - AO3.1b
	Max 10 marks
3	**8-10 marks** The candidate has: • justified the choice of areas to be included in the prototype system and explained the reasons for omitting the remaining areas • created a comprehensive range of screens and outputs for the chosen areas of the solution • created a functioning system that carries out all chosen processes using realistic data and fulfils all requirements for data output and storage • evaluated the functioning of the prototype and justified the good features of the prototype solution • described the shortcomings and made specific suggestions for improvement.
2	**4-7 marks** The candidate has: • described the areas to be included in the prototype system • created mock ups of screens and outputs related to all areas described • created a system that carries out most of the chosen system processes using realistic data to create required outputs • considered the functioning of the prototype system and described the good features and identified the shortcomings and made suggestions for some improvements.
1	**1-3 marks** The candidate has: • identified some areas of the proposed solution that are included in the prototype solution • created mock ups of the main screens and essential outputs • simulated some systems processes but does not use real data.
0	**0 marks** Response not credit worthy or not attempted.

Band	Post-Prototype Refinement of Design - AO3.1a
	Max 5 marks
3	**4-5 marks** The candidate has: • obtained feedback on the prototype system from a competent third party • fully described the feedback received and explained the implications of the feedback for the re-design of the system • re-considered and re-structured input and output facilities as necessary • given full consideration to the need for additional data as necessary • given full consideration to all required files and/or data structures in light of any need to include additional data • given full consideration to the need for any additional processes needed and has fully described any new relationships between the data and the processes of the revised design • fully described all processing routines using an appropriate recognised convention in sufficient detail for implementation by a competent third party • fully refined with detailed design, in response to feedback or justified instances where suggestions have been discounted.
2	**3 marks** The candidate has: • obtained feedback on the prototype system from a competent third party • described the feedback and identified the areas of the system to be refined • re-designed the input and output facilities in response to feedback if necessary • changed files and/or data structures as indicated from feedback • included additional processes to provide a working solution to the problem as necessary • revised existing processing routines and presented them using a recognised convention • included additional processing routines as necessary • provided an explanation where feedback suggestions have be discounted.
1	**1-2 marks** The candidate has: • obtained feedback on the prototype system from a competent third party • included a limited summary of the feedback • made limited alterations to outline designs for the input and output facilities as indicated from feedback • made limited amendments to the main files and/or data where necessary • attempted to re-define the routines using a recognised convention where necessary.
0	**0 marks** Response not credit worthy or not attempted.

Band	Software Development – AO3.1b
	Max 25 marks
4	**19-25 marks** The candidate has: • taken full account of the revised detailed design and refined the prototype in light of feedback and changed designs • produced a functioning solution to a highly demanding problem that meets most of the objectives for the solution to the problem. Better solutions will meet almost all of the objectives for the proposed solution. • used and fully exploited the programming facilities of the language • demonstrated a sound understanding of the appropriate tools and techniques available to them • created a well-structured data model that aids efficient data handling. Better candidates will present a data model normalised to third normal form. • produced a solution that is well-structured and modular in nature. The solution makes good use of local variables and minimises the use of global variables. • written code that is fully self-documenting and well-structured with annotation that will allow a competent third party to maintain the solution in future • made use of complex user-defined routines. Better candidates will have made effective use of recursive algorithms. • produced evidence of the effective use of validation for all key components. Better candidates will have created efficient routines for exception handling. • fully documented the variables and actual data structures used to create the solution to the chosen problem • included evidence of the completed user interface including a full description of the features that make it fit for audience and purpose.
3	**13-18 marks** The candidate has: • taken account of the reworked design and improved or extended the prototype in light of feedback and changes to designs • produced a functioning system to a demanding problem that meets many of the objectives for the solution of the problem. Better solutions will meet more of the central requirements. • used and exploited the programming facilities of the language • demonstrated an understanding of the tools and techniques available to them • created a data model involving a series of linked data structures to facilitate the manipulation of data and reduce data redundancy. Better candidates will have made use of multi-dimensional arrays. • produced a solution that is modular in nature that makes good use of local variables • written code that is self-documenting, Better candidates will have annotated all key components of their solutions. • made use of user-defined routines Better candidates will have used data handling routines such as sorts and searches. • provided evidence of effective validation for most key elements • used appropriate self-documenting identifiers for most variables and structures • provided evidence of the completed user interface with a description of the features used to make it fit for audience and purpose.

Band	Software Development – AO3.1b
	Max 25 marks
2	**7-12 marks** The candidate has: • improved the prototype in light of some of the feedback, implementing some of the features of the revised design • produced a partially functional solution to the problem that satisfies a range of the basic objectives for the proposed solution • used a range of the programming facilities of the language • demonstrated an understanding of the more important tools and techniques available to them • created a simple data model involving several linked data structures. Better candidates will have made use of arrays of at least two dimensions. • produced a solution with a clear structure. The code includes appropriate annotation for the majority of key components. • made use of routines such as linear searches and mathematical calculations • provided evidence of the use of basic validation • used appropriate identifiers for many variables and data structures. Better candidates will have made use of self-documenting identifiers. • provided evidence of the completed user interface with a description of the features used to make it largely fit for audience and purpose.
1	**1-6 marks** The candidate has: • reworked the prototype in light of some of the feedback. • produced a partially functional solution to the problem that satisfies some of the basic objectives for the proposed solution. • used some of the programming facilities of the language but has demonstrated a limited understanding of the tools and techniques available to them • made use of basic data structures with simple data types. Better candidates will have made use of single dimension arrays. • produced a solution that may be linear with some efficient code. Better candidates may have included some appropriate annotation of code. • provided little or no evidence of validation attempted to document the variables and actual data structures. Better candidates may have made an attempt to use appropriate identifiers. • provided evidence of the user interface with a limited description of its features
0	**0 marks** Response not credit worthy or not attempted.

Band	Developmental Testing – AO3.1c
	Max 5 marks
3	**4-5 marks** The candidate has: • provided evidence of comprehensive testing at each stage of the development of the solution to the chosen problem • provided evidence of all problems encountered and fully justified and evidenced the actions taken to overcome these problems.
2	**3 marks** The candidate has: • provided evidence of testing at most stages of the development of the solution to the chosen problem • provided evidence of problems encountered during development and described and evidenced the actions taken to overcome these problems.
1	**1-2 marks** The candidate has: • provided limited evidence of testing during the development of the solution to the chosen problem • provided limited evidence of problems encountered during development and the actions taken to remedy the errors in coding.
0	**0 marks** Response not credit worthy or not attempted.

Band	Testing – AO3.1c
	Max 10 marks
3	**8-10 marks** The candidate has: • produced a comprehensive plan for testing all areas of the system • made use of a wide range of appropriate testing methods • made use of test data including typical, extreme and invalid data where appropriate • presented all results with detailed and informed commentaries and included specific suggestions to refine the system.
2	**4-7 marks** The candidate has: • produced a test plan for testing most areas of the system • made use of a range of testing methods • made use of test data to test all parts of the system • presented results with suitable commentaries and suggestions for possible improvements to the system.
1	**1-3 marks** The candidate has: • produced a test plan for testing the main functions of the system • made use of more than one type of data in the testing process • presented results with comments and made limited suggestions for possible improvements to the system.
0	**0 marks** Response not credit worthy or not attempted.

Band	Evaluation – AO3.1c
	Max 15 marks
3	**11-15 marks** The candidate has: • produced a detailed evaluation of the effectiveness of the programming language and justified the tools and techniques used • compared and contrasted the completed system with similar commercially available systems • identified good features and shortcomings of the completed solution and described significant potential improvements that could be made to improve its effectiveness • evaluated their own strengths and weaknesses in the design and development of the system • described specific changes of approach that would be adopted in future to avoid problems experienced during the project • produced a well-structured and clearly expressed review. Specialist terms have been used with ease and accuracy. Work is error free.
2	**6-10 marks** The candidate has: • described the effectiveness of the tools and features of programming language used • compared the completed system with similar commercial available systems • identified the main good features of the solution and described appropriate potential improvements • described strengths and weaknesses in own performance in the design and creation of the system • produced an account of problems arising out of the project work and suggested strategies for improvement their own performance • produced a review that communicates meaning. Technical vocabulary has been used accurately. The work contains few errors.
1	**1-5 marks** The candidate has: • produced a limited description of the tools and features of the programming language used • made limited comparisons with a similar commercial system • identified several possible improvements • produced an account of own performance in the design and creation of the system • produced an account of problems arising out of the project work • produced a review that conveys meaning with limited technical detail. There is limited use of specialist vocabulary. The work may contain inaccuracies.
0	**0 marks** Response not credit worthy or not attempted.

APPENDIX C

Mathematical skills

Computer science uses mathematics to express its computational laws and processes.

This specification in computer science contains a minimum of 10% mathematics as required by the regulator. Students are asked to demonstrate their knowledge and understanding and skills of computational processes and problem-solving in both theoretical and practical ways. The following list shows the key concepts that will be common to all specifications in computer science.

For each topic below, while the concepts are Level 2 (though not all appear in GCSE Mathematics specifications), students will, however, be expected to apply the skills they acquire in a Level 3 context.

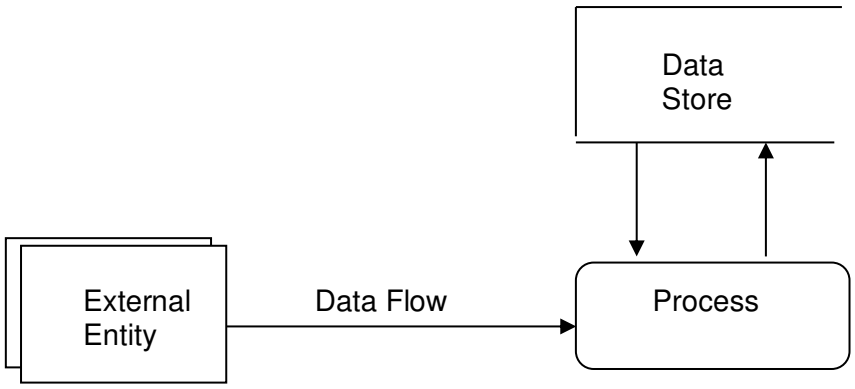
Topics:

- Boolean algebra
- Comparison of complexity of algorithms
- Number representations and bases

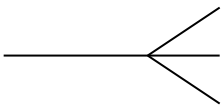
APPENDIX D -
Conventions followed in specification

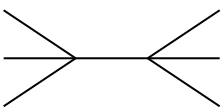
Note 1
In general, the diagram conventions used will be those described in the current edition of *The BCS Glossary of ICT and Computing Terms* (published BCS Learning and Development Ltd).

In the case of data flow diagrams, where no generally accepted symbols currently exist, candidates should be familiar with the following symbols, used in a number of current GCE textbooks:

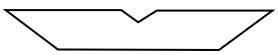


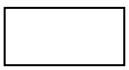
Note 2
The following symbols are used for entities and relationships.

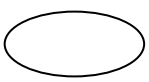
One-To-Many Relationship 

Many-To-Many Relationship 

Note 3
Where Von Neumann architecture is represented diagrammatically, the following symbols are used.

Arithmetic logic unit 

Register 

Control unit 

Note 4

Structured Query Language.

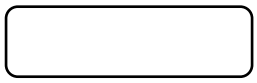
Candidates should be familiar with the following commands and operators:

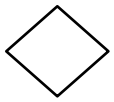
- CREATE TABLE …
- PRIMARY KEY
- NOT NULL
- Int
- Char(n)
- Numeric(m,n)
- DateTime
- INSERT INTO … VALUES
- SELECT … FROM … WHERE …
- SELECT * FROM … WHERE …
- IN
- AND
- OR
- ORDER BY
- GROUP BY
- UPDATE … SET …
- =
- >
- >=
- <
- <=
- <>

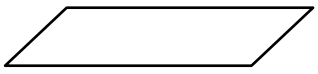
Candidates should also be familiar with the use of sub queries and parentheses.

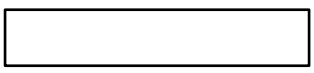
Note 5

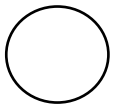
The following symbols are used in flowcharts:

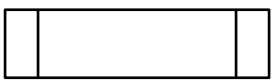
Start / Stop procedure 

Decision box 

Input / Output 

Operation 

Connector 

Store / Subroutine call 

Flow of control