GCSE

WJEC GCSE in COMPUTER SCIENCE

APPROVED BY QUALIFICATIONS WALES

SPECIFICATION

Teaching from 2017
For award from 2019

This Qualifications Wales regulated qualification is not available to centres in England.

WJEC GCSE in COMPUTER SCIENCE

For teaching from 2017
For award from 2019

This specification meets the Approval Criteria for GCSE Computer Science and the Approval Criteria for GCSE Qualifications which set out the requirements for all new or revised GCSE specifications developed to be taught in Wales from September 2017.

© WJEC CBAC Ltd.

GCSE COMPUTER SCIENCE (Wales)

SUMMARY OF ASSESSMENT

Unit 1: Understanding Computer Science **Written examination: 1 hour 45 minutes** **50% of the qualification** — **100 marks**
This unit investigates hardware, logical operations, communication, data representation and data types, operating systems, principles of programming, software engineering, program construction, security and data management and the impacts of digital technology on wider society.
Unit 2: Computational Thinking and Programming **On-screen examination: 2 hours** **30% of the qualification** — **60 marks**
This unit investigates problem solving, algorithms and programming constructs, programming languages, data structures and data types and security and authentication.
Unit 3: Software Development **Non-exam assessment: 20 hours** **20% of qualification** — **80 marks**
This unit requires learners to produce a programmed solution to a problem. They must analyse the problem, design a solution to the problem, develop a final programmed solution, test the solution and give suggestions for further development of the solution. Throughout the production of the solution learners are required to produce a refinement log that evidences the development of the solution.

This linear qualification will be available in May/June each year. It will be awarded for the first time in summer 2019.

Qualification Approval Number: C00/1157/9

© WJEC CBAC Ltd.

GCSE COMPUTER SCIENCE

1 INTRODUCTION

1.1 Aims and objectives

The WJEC GCSE in Computer Science encourages learners to:

- understand and apply the fundamental principles and concepts of computer science, including; abstraction, decomposition, logic, algorithms, and data representation
- analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs to do so
- think creatively, innovatively, analytically, logically and critically
- understand the components that make up digital systems, and how they communicate with one another and with other systems
- understand the impacts of digital technology to the individual and to wider society
- apply mathematical skills relevant to computer science.

Computers are widely used in all aspects of business, industry, government, education, leisure and the home. In this technological age, a study of computer science, and particularly how computers are used in the solution of a variety of problems, is essential to learners.

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 learners' horizons beyond the school or college environment in the appreciation of the effects of computer science on society and individuals.

The WJEC GCSE in Computer Science has been designed to give an 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.

This specification also gives learners an opportunity to produce extended written responses and demonstrate the quality of their written communication, including appropriate use of punctuation and grammar.

© WJEC CBAC Ltd.

1.2 Prior learning and progression

There are no previous learning requirements for this specification. Any requirements set for entry to a course based on this specification are at the school/college's discretion.

This specification builds on subject content which is typically taught at key stage 3 and provides a suitable foundation for the study of Computer Science at either AS or A level. 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.

1.3 Equality and fair access

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. candidates are allowed access to a Sign Language Interpreter, using British Sign Language). 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). As a consequence of provision for reasonable adjustments, very few learners will have a complete barrier to any part of the assessment.

© WJEC CBAC Ltd.

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 Skills Challenge Certificate within 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 must 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.

© WJEC CBAC Ltd.

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.

The subject content for GCSE Computer Science will be assessed across three units. Whist there is a degree of overlap between the content in Unit 1 and Unit 2, the context in which this content is assessed differs. In Unit 1, content is assessed in a theoretical way, whereas in Unit 2 it is assessed through its use within programs and algorithms.

Unit 1: Understanding Computer Science
Written examination: 1 hour 45 minutes
50% of the qualification
100 marks

Unit 2: Computational Thinking and Programming
On-screen examination: 2 hours
30% of the qualification
60 marks

Unit 3: Software Development
Non-exam assessment: 20 hours
20% of qualification
80 marks

© WJEC CBAC Ltd.

2.1 Unit 1

1. Hardware	
Architecture	Describe the characteristics of CPU architecture, including Von Neumann architectures. Identify and explain the role of the components of the CPU in the fetch-decode-execute cycle. Explain how performance is affected by the cache size, clock speed and number of cores. Explain the difference between RISC and CISC types of processors.
Input/output	Describe the use and characteristics of input and output devices.
Primary storage	Explain the functional characteristics of Random Access Memory (RAM), Read Only Memory (ROM), flash memory and cache memory.
Secondary storage	Describe the characteristics of contemporary secondary storage technologies including magnetic, optical and solid state. Explain the functional characteristics of contemporary secondary storage devices in terms of suitability, durability, portability and speed.
Storage requirements	Describe the relationship between data storage units, including bit, nybble, byte, kilobyte and additional prefix multipliers. Describe data capacity and calculate data capacity requirements.
Other hardware components	Describe the characteristics and role of other hardware, including GPU, sound cards and motherboards.
Embedded systems	Describe the use and give examples of embedded systems.

© WJEC CBAC Ltd.

2. Logical operations	
Logical operators	Use `AND`, `OR`, `NOT` and `XOR` logical operators, combinations of these, and their application in appropriate truth tables to solve problems.
Boolean logic	Simplify Boolean expressions using Boolean identities and rules.

3. Communication	
Networks	Explain the characteristics of networks and the importance of different network types, including LAN and WAN. Describe the importance of common network topologies, including ring, star, bus and mesh, and their advantages and disadvantages. Explain the importance of connectivity, both wired and wireless. Explain and give advantages and disadvantages of circuit switching and packet switching. Explain the importance and the use of a range of contemporary network protocols, including Ethernet, Wi-Fi, TCP/IP, HTTP, HTTPS, FTP and email protocols. Describe the typical contents of a TCP/IP packet. Explain the importance of layers and the TCP/IP 5-layer model. Describe methods of routing traffic on a network and calculate routing costs.
Internet	Explain how Domain Name System (DNS) servers and Internet Protocol (IP) addresses work.

© WJEC CBAC Ltd.

4. Organisation and structure of data	
Representation of numbers	Use and convert between denary, binary (up to 16 bits) and hexadecimal counting systems. Explain the use of hexadecimal notation as shorthand for binary numbers. Use arithmetic shift functions and explain their effect. Apply binary addition techniques. Explain the concept of overflow.
Representation of graphics and sound	Explain the digital storage of graphics. Explain the digital storage and sampling of sound. Describe the use of metadata in files.
Storage of characters	Describe how characters are stored as a binary number. Describe standardised character sets, including Unicode and American Standard Code for Information Interchange (ASCII).
Data types	Describe the concept of data types, including integer, Boolean, real, character and string.
Data structures	Describe, design, interpret and manipulate data structures including records, one-dimensional and two-dimensional arrays. Select, identify and justify appropriate data structures for given situations.
File design	Design files and records appropriate for a particular application.
Data validation and verification	Explain and use appropriate techniques for data validation and verification. Design algorithms and programming routines that validate and verify data.

© WJEC CBAC Ltd.

5. System software	
Managing resources	Describe the purpose and functionality of the operating system in managing resources, including peripherals, processes, memory and backing store.
Providing an interface	Describe the purpose and functionality of the operating system in providing a user interface.
Utility software	Explain the purpose and functionality of a range of utility software.

6. Principles of programming	
Levels of computer language	Describe the characteristics and purpose of high-level and low-level languages. Identify and describe situations that require the use of a high-level or a low-level language.

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

8. Program construction	
Compilers, interpreters and assemblers	Describe the purpose and give examples of the use of compilers, interpreters and assemblers. Explain the principal stages involved in the compilation process: lexical analysis, symbol table construction, syntax analysis, semantic analysis, code generation and optimisation. Describe and give examples of programing errors.

© WJEC CBAC Ltd.

9. Security and data management	
Data security	Describe the dangers that can arise from the use of computers to store personal data. Describe methods that protect the security of data including access levels, suitable passwords for access and encryption techniques.
Data management	Explain the need for file backups and generations of files. Explain the need for archiving files.
Compression	Explain how lossy and lossless data compression algorithms are used. Calculate compression ratios.
Network security	Recognise the importance of network security and describe the dangers that can arise from the use of networks. Explain the purpose and typical contents of an acceptable use policy and disaster recovery policy.
Cybersecurity	Describe the characteristics and explain the methods of protection against malware, including viruses, worms and key loggers. Describe the different forms of attack based on technical weaknesses and/or user behaviour. Describe methods of identifying vulnerabilities. Explain different ways of protecting software systems during design, creation, testing and use. Describe the role of internet cookies.

10. Ethical, legal and environmental impacts of digital technology on wider society	
Ethical	Describe the ethical impacts of digital technology, including issues of privacy and cybersecurity. Explain the importance of conforming to professional standards, including formal and informal codes of ethical behaviour.
Legislation	Explain how relevant current legislation impacts on security, privacy, data protection and freedom of information.
Environmental issues	Describe the environmental impacts of digital technology on wider society.

© WJEC CBAC Ltd.

2.2 Unit 2

1. Problem solving	
Problem solving	Use a systematic approach to problem solving including the use of decomposition and abstraction. Use abstraction effectively to model selected aspects of the external world in an algorithm or program. Use abstraction effectively to appropriately structure programs into modular parts with clear, well-documented interfaces.

2. Algorithms and programming constructs	
Algorithms	Use common methods of defining algorithms, including pseudo-code and flowcharts *(see Appendix C)*.
Programming constructs	Identify, explain and use sub routines in algorithms and programs. Identify, explain and use sequence, selection and iteration in algorithms and programs. Identify, explain and use counts and rogue values in algorithms and programs. Identify and explain constructs in object orientated programs. Follow and make alterations to algorithms and programs that solve problems using: • sequence, selection and iteration • input, processing and output. Write algorithms and programs that solve problems using: • sequence, selection and iteration • input, processing and output.
Variables	Identify, explain and use local and global variables in algorithms and programs.
Identifiers	Explain why the use of self-documenting identifiers and annotation are important in programs. Give examples of self-documenting identifiers and annotation.

© WJEC CBAC Ltd.

2. Algorithms and programming constructs (cont.)	
String handling	Identify, explain and use routines for string handling in algorithms and programs.
Mathematical operations	Identify, explain and apply computing-related mathematical operations in algorithms and programs (*see Appendix C*).
Logical operations	Identify, use and explain the logical operators `AND`, `OR`, `NOT` and `XOR` in algorithms and programs *(see Appendix C)*.
Sorting	Describe the characteristics of merge sort and bubble sort algorithms.
Searching	Explain and use linear and binary search algorithms.
Testing and evaluation	Explain how an algorithm or program works and evaluate its fitness for purpose in meeting requirements. Evaluate the efficiency of an algorithm or program using logical reasoning and test data.

3. Programming languages	
Markup languages	Design, write, test and refine HTML pages using the following tags: • HTML `<html>` • Head `<head>` • Title `<title>` • Body `<body>` • Headings `<h1>-<h6>` • Paragraph `<p>` • Italic `<i>` • Bold `<b>` • Centre align `<center>` • Anchor `<a href="URL">` • Mailto `<a href="mailto:address">` • Unordered List `<ul>` • List Item `<li>` • Blockquote `<blockquote>` • Horizontal Rule `<hr>` • Image `<img>` • and their corresponding closures.

© WJEC CBAC Ltd.

3. Programming languages (cont.)	
Object oriented languages	Design, write, test and refine Java programs within the Greenfoot environment, using the following skills: • Create new and extend existing classes • Create new and edit existing objects • Create new and edit existing worlds • Write and invoke methods • Change existing methods • Create new and edit existing properties (including public, private, static, etc.) • Add and remove objects from worlds • Use actors • Move objects around a world • Keyboard input • Add and play sounds • Implement and use parameter passing (by value and by reference) • Access one object from another • Implement object collision detection • Implement random number generation • Use the concept of inheritance and encapsulation.
Assembly language	Design, write, test and refine simple assembly programs using the following mnemonics: • Input `INP` • Output `OUT` • Store `STA` • Load `LDA` • Add `ADD` • Subtract `SUB` • Branch `BRA` • End/Stop/Halt `HLT` • Data definition `DAT`

© WJEC CBAC Ltd.

4. Data structures and data types	
Implementing data structures	Use one-dimensional and two-dimensional arrays, files and records.
Implementing data types	Use a variety of data types, including integer, Boolean, real, character and string.
Variables and constants	Assign, identify and explain the use of constants and variables in algorithms and programs. Describe the scope and lifetime of variables in algorithms and programs.

5. Security and authentication	
Security techniques	Use appropriate security techniques, including validation and authentication.

© WJEC CBAC Ltd.

2.3 Unit 3

The non-exam assessment (NEA) is designed to assess a candidate's ability to apply the knowledge and understanding gained from Units 1 and 2. Candidates will be presented with a given scenario describing the requirements for a computer based solution. All work carried out for Unit 3 should be under teacher supervision, with no access to the Internet or email.

Candidates will be required to create a solution to the given scenario using the logical and systematic approach outlined below. Candidates will be required to analyse the given scenario, design, implement and test a solution to the given scenario and identify future developments that could be used to refine the outcome.

In addition candidates will be required to keep a refinement log as described below, explaining any issues encountered and resulting refinements to the original design.

Software Development	
Scope of the problem	Analyse the given scenario in terms of input, processing and output. Set objectives, including measurable success criteria for the proposed system.
Design	Design and document the input and output facilities required to produce an effective user interface. Design and document all required data structures. Using a standard convention such as pseudo code or flowcharts, design and document the following routines: • validation • data handling and processing • authentication.
Refinement Log	Plan to carry out activities in an appropriate order. Evaluate the progress made in each session. Describe any problems encountered using appropriate technical language. Justify any changes made to the original design as a result of problems encountered. Produce a logical plan of action for further sessions.
Effectiveness of solution	Create a solution that fulfils the requirements of the given scenario. Create an intuitive interface that is fit for audience and purpose. Ensure that the solution is well-structured and modular in nature. Ensure that the solution makes effective use of resources and is secure, robust and reliable. Make use of authentication routines.

© WJEC CBAC Ltd.

Software Development (cont.)	
Technical quality	Create code for the solution that is self-documenting, uses meaningful identifiers and appropriate constants. Use a program style that is consistent, including indentation and appropriate use of white space. Use subroutines with well-defined interfaces. Use local and global variables. Create validation and exception handling routines. Annotate code so that it is accessible to a competent third party.
Test strategy	Design and document an effective testing strategy that will ensure that the final solution meets the requirements of the given scenario. Describe the purpose of unit, integration and functional testing. Describe how the outcomes of the testing process can be used to inform further development of the solution. Design a comprehensive plan for carrying out unit, integration and functional testing to cover all the requirements of the given scenario. Design test data to include examples of typical, extreme and erroneous content.
Testing	Implement the comprehensive test plan using typical, extreme and erroneous data where appropriate. Present all test outcomes with detailed and informed commentaries.
Further development	Consider the outcomes of the testing process in terms of the system objectives. Describe successful features and areas for improvement. Propose suggestions for specific extensions to the solution.

Please refer to the banded mark scheme on pages 29 to 35 for full details of assessment criteria.

© WJEC CBAC Ltd.

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 key concepts and principles of computer science

AO2
Apply knowledge and understanding of key concepts and principles of computer science

AO3
Analyse problems in computational terms to make reasoned judgements and to design, program, evaluate and refine solutions

The table below shows the weighting of each assessment objective for each unit and for the qualification as a whole.

	AO1	AO2	AO3
Unit 1	26%	24%	-
Unit 2	4%	14%	12%
Unit 3	-	2%	18%
Total	**30%**	**40%**	**30%**

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 and Vocational Qualifications: Instructions for conducting examinations* (Joint Council for Qualifications). This document is available on the JCQ website (www.jcq.org.uk).

Programming language for Unit 2

Some tasks in Unit 2 will require work to be completed using Greenfoot, which is an integrated development environment (IDE) freely available for legal download. (http://www.greenfoot.org/door)

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

© WJEC CBAC Ltd.

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 Greenfoot pre-installed
- file(s) supplied by WJEC that have been pre-installed and tested.

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 3 non-exam assessment

In this specification, the non-exam assessment is a project. This is a substantial piece of work, undertaken over an extended period of time.

The project requires candidates to analyse the scope of a problem, design a solution to the problem, implement a solution, design a test strategy, test the solution and give suggestions for further development of the solution. In addition candidates will be required to keep a refinement log as described in Unit 3, explaining any issues encountered and resulting refinements to the original design.

The task will be set annually by WJEC and published on the WJEC secure website. The task will be released each September, from September 2017, and will be accepted for submission in the year specified only, as shown in the table below[1].

	Year of award			
Task release date	2019	2020	2021	2022
September 2017	✓	×	×	×
September 2018	×	✓	×	×
September 2019	×	×	✓	×
September 2020	×	×	×	✓

Candidates will develop a piece of work over 20 hours; produce a solution to a given problem and a word-processed report with an advisory limit of 2000 words detailing the work carried out. The 2000 word count does not include the program code or annotation.

[1] As this is a linear qualification, all assessments must be taken at the end of the course.

© WJEC CBAC Ltd.

The work for Unit 3 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.

WJEC will support the following languages:

- Basic derived languages (e.g. VB.NET, Small Basic, QBasic)
- C derived languages (e.g. C, C++, C#)
- PHP
- Python
- Pascal/Delphi

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 3, teachers should study the assessment grids in Appendix B, which are designed to present a system that links the assessment objectives to marks, and help 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 should make specific reference to the assessment criteria contained in the assessment grids in the comments they write on the work and on the cover sheets.

Submission of assessments

Candidate work for Unit 3 must be submitted to WJEC electronically. The submission should include a functioning compiled version and a functioning uncompiled copy of the solution and supporting documents in portable document format (pdf).

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 folders are organised in such a way that will allow the assessor and moderator to access the relevant pdf documents.

© WJEC CBAC Ltd.

The centre assessor should store the candidate's assessment documentation using the same naming convention as above, but with the addition of CS3. For example, Diane Smith's assessment documentation would be stored as CS3_68999_12345_SM_D.

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:

- 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. 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, a sanction may be applied. 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 3, all work must be undertaken under the direct supervision of teachers. Candidates must work in a dedicated, 'clean' account, with no access to the internet or email.

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 that she/he is entirely satisfied that the work submitted is that of the candidate concerned.

Centres entering candidates for GCSE 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.

The time spent working on the project for Unit 3 should be recorded by the teacher as a log and this may be requested by WJEC in addition to the work submitted. The log should be monitored continually by the centre and the candidate should not be allowed to exceed the time limit permitted for this project.

© WJEC CBAC Ltd.

Collaboration control

Candidates must not work together in the development of their project. Should workstations be near each other in class, the supervising teacher must ensure that all work carried out by candidates is done so independently.

Help files

Only the help files native to the programming language should be provided, where help files are online (e.g. Visual Basic.NET) the relevant help file image may be installed locally for access by the candidate. Forums or sample code should not be made accessible to candidates.

Candidates may also use a clean copy of the pseudocode conventions used by WJEC (Specification Appendix C, Note 3). This can be installed on the candidate's secure area. Where a candidate wishes to use a hard copy of these conventions, these should be handed to the candidate at the beginning of any session and collected at the end of the session. Notes may not be taken into or out of the controlled test session.

Should a centre wish to provide a written glossary to the syntax of a language, this is permitted, but it should only provide a high level guide on the use of syntax of the programming language. It must not give examples of segments of code or how to use the language. This can be installed on the candidate's secure area. Where a candidate wishes to use a hard copy of these glossaries, these should be handed to the candidate at the beginning of any session and collected at the end of that session. Glossaries must not be taken into or out of the controlled test session and must be submitted with the candidate's work.

Use of libraries

Candidates should avoid the use of pre-compiled units, libraries or modules in their project and should aim to construct the entire project from original code.
Where a library is used it must be pre-compiled. The candidate should state the functionality from the library to be used in their project. The supervising teacher should only permit candidates to use functionality that lies beyond the scope of a GCSE candidate (for example a library function that sorts data would not be acceptable as this functionality is within the scope expected from a GCSE candidate). The supervising teacher must ensure that candidates do not use any additional functions beyond those permitted.

The candidate must record the source of the library in their report and the function(s) used and supply an electronic version of the library with their program.

The supervising teacher must record details of the library and functions used in the candidate declaration.

© WJEC CBAC Ltd.

Feedback control

Supervising teachers may only provide support if a candidate's work is such that it prevents them from going on to the next part of the project. Supervising teachers must not give advice to candidates on how to improve their work if their work is sufficient in one part to permit the next part of the project to be developed. Only certain parts therefore are considered suitable for feedback.

Scope of the problem

The supervising teacher must not break down the requirements or decompose the problem for the candidate but may give support to identify sufficient high-level data, processing and outputs to continue to the next part of the project.

Design

The supervising teacher may give support to develop a minimally functional solution that may not satisfy all requirements of the given task, to allow the candidate to move on to the next part of the project.

Refinement log

Feedback must not be provided.

Effectiveness of solution

The supervising teacher must not provide any program code for the solution. The supervising teacher may help candidates in high level terms of the syntax within a programming language. The supervising teacher must not however give examples of how to build segments of programmed code. For example, giving an example of an implementation of an IF.. THEN.. ELSE statement would not be appropriate.

Technical quality

Feedback must not be provided.

Test strategy

The supervising teacher may give a high level indication of the areas of the program that need to be tested, however the supervising teacher must not give hints on how to test, provide any example test data, or specify that typical, extreme and erroneous data should be used.

Testing

Feedback must not be provided.

Further development

Feedback must not be provided.

© WJEC CBAC Ltd.

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 only 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 April 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 candidates 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 May.

© WJEC CBAC Ltd.

4 TECHNICAL INFORMATION

4.1 Making entries

This is a linear qualification in which all assessments must be taken at the end of the course. Assessment opportunities will be available in the summer series each year, until the end of the life of this specification. Summer 2019 will be the first assessment opportunity.

Where candidates wish to re-sit the qualification, both Units 1 and 2 must be retaken, whereas marks for Unit 3 may be carried forward. Results for Unit 3 have a shelf-life limited only by the shelf-life of the qualification.

The entry codes appear below.

Qualification title	Entry codes	
	English-medium	Welsh-medium
WJEC GCSE Computer Science	3500QS	3500CS

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

4.2 Grading, awarding and reporting

GCSE qualifications are reported as a grade on the scale from A* to G, where A* is the highest grade. Results not attaining the minimum standard for the award will be reported as U (unclassified).

© WJEC CBAC Ltd.

APPENDIX A

Non-exam assessment

Unit 3 – Software Development Task

Given Task
The task for Unit 3 will be a scenario set by WJEC and will take the form of a project that will be available for submission only in the year specified. A different scenario will be set for each academic year. All work carried out for this project should be under teacher supervision, with no access to the Internet or email. The time permitted for this project is 20 hours, and all time spent on the project should be monitored and logged by the centre as detailed in this specification.

The scenario will provide candidates with a description of a client's need for a new computer based solution to a given problem. In addition to a functional computer program, candidates will need to produce a word processed report with an advisory limit of 2000 words (not including annotated code or the refinement log). The areas for inclusion in the report are covered in detail in the content for this unit and are summarised below:

Scope of the problem
- Analysis of the given scenario in terms of input, processing and output
- Objectives, including measurable success criteria for the proposed system

Design
Descriptions of:
- Input and output facilities required to produce a user interface
- Data structures that will be required
- Documentation of the following routines using a standard convention (pseudo code or flowchart):
    - validation routines
    - data handling and processing
    - authentication

Software development (incorporating effectiveness of solution and technical quality)
- Annotated listing(s) of all programming code
- Evidence of the user interface

Test strategy
- Description of the test strategy
- Description of the purpose of unit, integration and functional testing
- Description of how outcomes inform further development
- Test plan and test data

Testing
- Evidence of implementation of the test plan and test outcomes with commentaries

Further development
- Discussion of the outcomes of the testing
- Description of the successful features of the solution and identification of areas for further development
- Suggestions for extensions to the solution

© WJEC CBAC Ltd.

Refinement log

The refinement log is an integral part of the project and should be completed during each session. The purpose of the log is for candidates to demonstrate that they are working in a logical and systematic manner.

Candidates are expected to record any issues encountered and how these issues were addressed.

The refinement log is supplied as an electronic document and must be submitted with the computer program and the report.

A sample log page is shown below.

Session 3	
Date	
Length of session	
Progress made in this session	
Problems encountered with the project	
Changes made to original designs as a result of problems	
Action plan for next session	
Project plan status (on time, ahead of time, behind time)	
Action plan to manage time	

Please refer to the banded mark scheme on pages 29 to 35 for full details of assessment criteria.

© WJEC CBAC Ltd.

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 candidate'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 candidate'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 candidate'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 candidate'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 candidate'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 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 candidate'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.

© WJEC CBAC Ltd.

Unit 3 – Software Development Task

Band	Scope of the problem - AO3
	Max 8 marks

4	**7 - 8 marks**
	The candidate has:
	• Completed a thorough analysis of the given scenario identifying all:
	  ○ data required to create an effective solution
	  ○ processing to be carried out by the solution
	  ○ required outputs from the solution
	• Produced a detailed set of objectives, that are measurable, that define clearly the tasks required to create an effective and fully functional solution

3	**5 - 6 marks**
	The candidate has:
	• Completed an analysis of the given scenario, with no significant omissions, identifying most of the:
	  ○ data required to create a functional solution
	  ○ processing to be carried out by the solution
	  ○ required outputs from the solution
	• Produced a set of objectives, most of which are measurable, that define the tasks required to create a functional solution

2	**3 – 4 marks**
	The candidate has:
	• Carried out an analysis of the given scenario, identifying the basic:
	  ○ data required to create a working solution
	  ○ processing requirements to produce a working solution
	  ○ outputs from the solution
	• Produced a set of objectives, a minority of which are measurable, that describe the main tasks required to create a working solution

1	**1 - 2 marks** The candidate has: • Carried out a superficial analysis of the given scenario that has only partially identified the input, processes and output required to produce a working solution • Produced objectives that describe the main tasks required to create a working solution
	0 marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

Band	Design - AO3
	Max 12 marks
4	**10 - 12 marks** The candidate has: • Produced a comprehensive design that would allow a competent third party to create a solution that covers all stated objectives • Identified fully and described in detail the input and output facilities to be provided by the user interface which will be fit for purpose • Described all data structures required to create an effective solution, using correct technical terminology • Described fully the validation routines required to ensure that only appropriate data can be entered into the solution • Considered fully the need for authentication routines • Described all data handling and processing routines for an effective solution as algorithms, using a standard convention such as pseudo code or flowcharts
3	**7 - 9 marks** The candidate has: • Used the objectives for the solution to inform a design that will produce the facilities required to ensure that the solution is functional • Identified and described most of the input and output facilities to be provided by the user interface and has considered the needs of the user • Described most data structures required using appropriate terminology • Identified most inputs that will require validation and outlined proposals for implementing validation routines • Considered the need for authentication routines • Described most data handling and processing routines for the solution as algorithms, using a standard convention such as pseudo code or flowcharts
2	**4 - 6 marks** The candidate has: • Used the objectives for the solution as a basis for the design that will produce a solution that will achieve a majority of the required functionality • Identified the basic input and output facilities to be provided by the user interface • Identified the data structures required to produce a solution that is partially functional but carries out the basic requirements of the given scenario • Identified several inputs that will require validation • Outlined the need for authentication routines • Described the basic data handling and processing routines for the solution as algorithms, using a standard convention such as pseudo code or flowcharts. Some routines may be incorrect or incomplete.
1	**1 - 3 marks** The candidate has: • Used the objectives for the solution as a basis for an outline design for a partial solution • Produced outline designs for the identified input and output facilities provided by the user interface • Outlined the key files and/or data structures required to produce a partial solution • Given consideration to possible validation of input data, which may not be accurate or appropriate • Partially outlined data handling and processing routines that may use a standard convention such as pseudo code or flowcharts. The descriptions may not be accurate or correct.
	0 marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

Band	Refinement Log - AO3 **Max 5 marks**
3	**4 - 5 marks** The candidate has: • Demonstrated a structured approach to developing the solution • Carried out activities in an appropriate order • Evaluated effectively the progress made in each session • Provided a full description of any problems encountered with good use of technical terminology • Justified any changes that have been made to the original design demonstrating an informed understanding of the need for change • Produced logical and prioritised actions for subsequent sessions
2	**2-3 marks** The candidate has: • Demonstrated a structured approach to developing the solution • Carried out most activities in an appropriate order • Described the progress made in each session • Provided a description of any problems encountered with satisfactory use of technical terminology • Described any changes that have been made to the original design • Produced sensible actions for subsequent sessions
1	**1 mark** The candidate has: • Outlined the progress made in most sessions • Described problems encountered but may lack the use of technical terminology • Outlined changes that have been made to the original design • Identified one or more activity for the next session
	0 marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

Band	Effectiveness of solution – AO3
	Max 15 Marks
5	**13 – 15 Marks** The candidate has created a solution that: • Achieves all the requirements of the given scenario • Is usable with a user interface that is intuitive and fit for audience and purpose • Is well-structured and modular in nature • Is efficient in use of resources • Is secure with effective authentication routines • Is reliable and robust
4	**10 – 12 Marks** The candidate has created a solution that: • Achieves most of the requirements of the given scenario. • Is usable with a user interface that is functional and generally easy to use • Is structured and modular in nature • Is secure with authentication routines • Is generally robust
3	**7 – 9 Marks** The candidate has created a solution that: • Achieves a majority of the requirements of the given scenario • Provides evidence of a functional user interface • Is structured in nature • Makes use of authentication routines
2	**4 – 6 Marks** The candidate has created a solution that: • Achieves a minority of the requirements of the given scenario • Includes a basic user interface that may not be fit for purpose • Is partially structured in nature
1	**1 – 3 Marks** The candidate has created a solution that: • Partially achieves a minority of the requirements of the given scenario • Has a partially functional user interface
	0 Marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

Band	Technical Quality – AO3 Max 20 Marks
5	**17 – 20 Marks** The candidate has created a successful solution that covers all the requirements of the given scenario. The candidate has: • Written code that is self-documenting, well-structured and modular in nature • Used a consistent programming style throughout, including indentation and the use of white space around operators and keywords • Made full use of meaningful identifiers and appropriate use of constants • Created subroutines with well-defined interfaces • Made effective use of local variables and minimised the use of global variables • Produced effective validation routines and created routines for exception handling • Provided informed annotation of the code where appropriate
4	**13 – 16 Marks** The candidate has created a functional solution that covers most requirements of the given scenario. The candidate has: • Written code that is self-documenting and modular in nature • Used a consistent programming style including appropriate indentation • Made use of meaningful identifiers and appropriate use of constants • Made use of local variables and minimised the use of global variables • Produced validation routines and created routines for exception handling • Provided effective annotation of the code where appropriate
3	**9 – 12 Marks** The candidate has created a functional solution that covers the majority of the requirements of the given scenario. The candidate has: • Written code that is self-documenting • Used a consistent programming style including indentation • Made use of meaningful identifiers and constants • Made use of local variables and generally minimised the use of global variables • Produced validation routines • Provided annotation of the code where appropriate
2	**5 – 8 Marks** The candidate has created a solution that covers the basic requirements of the given scenario. The candidate has: • Written code that includes some self-documentation • Used indentation • Made use of meaningful identifiers • Provided basic annotation of the code

© WJEC CBAC Ltd.

Band	Technical Quality (cont.) – AO3
	Max 20 Marks
1	**1 – 4 Marks** The candidate has created a partial solution to at least one of the requirements of the given scenario. The candidate has: • Made use of meaningful identifiers • Used indentation
	0 Marks Response not credit worthy or not attempted

Band	Test strategy - AO2
	Max 8 marks
3	**6 - 8 marks** The candidate has: • Considered fully the nature of the solution when developing a well-structured test strategy • Provided an informed description of the scope and range of the chosen test strategy • Fully explained the purpose of unit, integration and functional testing, taking into account the nature of the solution • Considered in detail how the outcomes of the testing process will be used to influence any further development of the solution • Produced a comprehensive plan for carrying out unit, integration and functional testing to cover all requirements of the given scenario • Identified comprehensive test data to fully test the solution
2	**3 - 5 marks** The candidate has: • Considered the nature of the solution when developing a test strategy • Provided a description of the scope and range of the chosen test strategy • Explained the purpose of unit, integration and functional testing, taking into account the nature of the solution • Considered how the outcomes of the testing process may be useful in any further development of the solution • Produced a test plan to carry out unit, integration and functional testing of most of the requirements of the given scenario • Identified appropriate test data to test most functionality of the solution
1	**1 - 2 marks** The candidate has: • Attempted to explain the scope and range of the chosen test strategy • Identified the purpose of testing the solution • Produced a test plan to carry out the testing of the solution but the plan may not cover all key areas • Identified test data to partially test the solution
	0 marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

Band	Testing - AO3
	Max 8 marks
3	**6 - 8 marks** The candidate has: • Followed the test plan in a logical and systematic manner • Made effective use of typical, extreme and erroneous data • Presented all testing outcomes with detailed and informed commentaries • Produced writing which is very well structured using accurate grammar, punctuation and spelling.
2	**3 - 5 marks** The candidate has: • Used the test plan to carry out testing of the solution • Made use of realistic data to test all areas of the solution • Presented testing outcomes with suitably technical commentaries • Produced writing which is is generally well structured using reasonably accurate grammar, punctuation and spelling.
1	**1 - 2 marks** The candidate has: • Made use of data to test most areas of the solution that have been completed • Presented testing outcomes with brief correct commentaries • Produced writing which shows some evidence of structure but with some errors in grammar, punctuation and spelling.
	0 marks Response not credit worthy or not attempted

Band	Further development – AO3
	Max 4 marks
3	**4 marks** The candidate has: • Considered fully the outcomes of the testing process in terms of the solution objectives • Fully described the successful features and areas for further development • Proposed detailed and comprehensive suggestions for specific extensions to the solution
2	**2 - 3 marks** The candidate has: • Considered most outcomes of the testing processes against the solution objectives • Described most successful features and areas for further development • Proposed specific suggestions for extensions to the solution
1	**1 mark** The candidate has outlined: • Outcomes of the testing process • Suggestions for extension to the solution
	0 marks Response not credit worthy or not attempted

© WJEC CBAC Ltd.

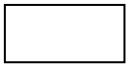
APPENDIX C

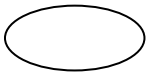
Conventions followed in specification

Note 1

Where Von Neumann architecture is represented diagrammatically, the following symbols are used:

Arithmetic logic unit 

Register 

Control unit 

Note 2

Where candidates are required to apply computing-related mathematics, the following arithmetical and relational operators will be used:

Operator	Meaning	Example
>	Greater than	A>B will return TRUE if the value of A is higher than the value of B otherwise it will return FALSE.
<	Less than	A<B will return TRUE if the value of A is lower that the value of B otherwise it will return FALSE.
<=	Less than or equal to	A<=B will return TRUE if A is the same as or lower than B otherwise it will return FALSE.
>=	Greater than or equal to	A>=B will return TRUE if A is the same as or higher than B otherwise it will return FALSE.
<>	Not equal to	A<>B will return TRUE if A is not the same as B but FALSE if A is the same as B.
EQUALS (usually ==)	The same as	A==B will return TRUE if A is the same as B otherwise it will return FALSE.
AND	Both statements must be true for the argument as a whole to be true.	(A == 1) AND (B==4) will return TRUE if A is 1 and B is 4. It would return FALSE in all other situations.
OR	Only one of the statements needs to be true for the argument as a whole to be true.	(A==1) OR (B==4) will return TRUE if A is 1 or B is 4. It would only return FALSE if A is not 1 and B is not 4.
NOT	The opposite of	NOT(A) will return TRUE if A is FALSE and FALSE if A is TRUE.

© WJEC CBAC Ltd.

XOR	The argument is false if both statements are true. The argument is false if both statements are false. Otherwise the statement is true.	A XOR B would return TRUE if A and B are different values.
DIV	Integer division Finds the quotient or the 'whole number of times' a divisor can be divided into a number.	11 DIV 2 = 5 The quotient is 5 as 2 divides into 11 a whole number of 5 times
MOD	Modulo division Finds the remainder when a divisor is divided into a number.	11 MOD 2 = 1 The remainder is 1 as 2 divides 5 times into 11 with '1 remaining'

Note 3

Algorithms written in pseudo code will be represented using the following convention:

Construct	Example usage
Declare subroutines	`Declare CapitalLetterOfName` `End Subroutine`
Call a subroutine	`call SubroutineNeeded`
Declare and use arrays	`myarray[99]`
Literal outputs	`output "Please enter a number"`
Variable names	`myvariable`
Define variable data type	`myvariable is integer`
Data types	`integer, character, string, boolean`
Assignment	`set counter = 0`
Selection	`if . . . else . . . end if`

Indent at least single space after if or do or repeat etc.	`if counter = 1`
	`  output counter`
	`end if`

Annotation	`{Some annotation goes here}`
Comments (for Java only)	`/** Comments for Java */`

Repetition	`for i . . . next i`
	`repeat . . . until`
	`do . . . loop`
	`do . . . while`
	`while . . . repeat`

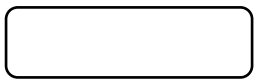
Logical operators `AND OR NOT XOR` will be in upper case.
Logical `TRUE` and `FALSE` will be in upper case.

© WJEC CBAC Ltd.

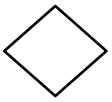
Note 4

Algorithms represented using a flowchart will use the following convention:

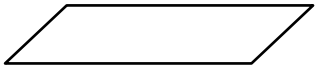
Start / Stop procedure



Decision box

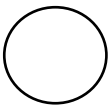


Input / Output

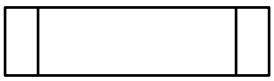


Operation

Connector



Store / Subroutine call



Flow of control
(Arrowhead indicates
direction of flow)

© WJEC CBAC Ltd.