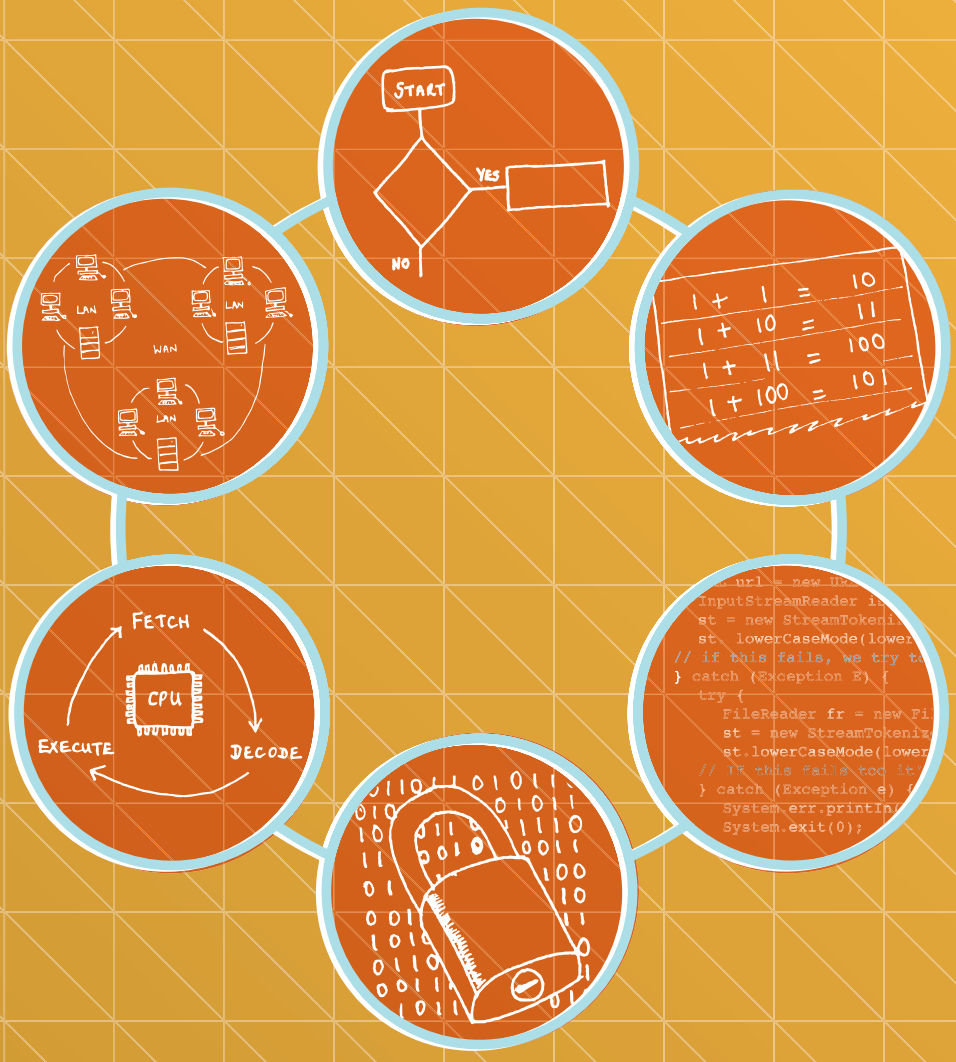
edexcel

PEARSON EDEXCEL GCSE in Computer Science



SPECIFICATION

Pearson Edexcel Level 1/Level 2 GCSE in Computer Science
First examination 2015

Issue 2

Edexcel, BTEC and LCCI qualifications

Edexcel, BTEC and LCCI qualifications are awarded by Pearson, the UK's largest awarding body offering academic and vocational qualifications that are globally recognised and benchmarked. For further information, please visit our qualification websites at www.edexcel.com, www.btec.co.uk or www.lcci.org.uk. Alternatively, you can get in touch with us using the details on our contact us page at www.edexcel.com/contactus

About Pearson

Pearson is the world's leading learning company, with 40,000 employees in more than 70 countries working to help people of all ages to make measurable progress in their lives through learning. We put the learner at the centre of everything we do, because wherever learning flourishes, so do people. Find out more about how we can help you and your learners at: www.pearson.com/uk

This specification is Issue 2. Key changes are sidelined. We will inform centres of any changes to this issue. The latest issue can be found on the Edexcel website: www.edexcel.com

References to third-party material made in this specification are made in good faith. We do not endorse, approve or accept responsibility for the content of materials, which may be subject to change, or any opinions expressed therein. (Material may include textbooks, journals, magazines and other publications and websites.)

Authorised by Martin Stretton
Prepared by Alex Martin

ISBN 978 1 446 91422 9

Introduction

The Pearson Edexcel Level 1/Level 2 GCSE in Computer Science is designed for use in schools and colleges. It is part of a suite of GCSE qualifications offered by Pearson.

About this specification

Rationale

The Pearson Edexcel Level 1/Level 2 GCSE in Computer Science has been developed in response to a number of recent initiatives aimed at promoting computer science as a rigorous, knowledge-based subject discipline that should be part of every young person's education.

These initiatives include:

- Recommendation 7 of the Royal Society report 'Shut down or restart? The way forward for computing in UK schools' (January 2012)
- 'Computer Science: A curriculum for schools' produced by the Computing at School (CAS) Working Group (March 2012)
- 'Computing: Programmes of study for Key Stages 1–4' (draft) published by the Department for Education (July 2013)

Content

The content of the Pearson Edexcel Level 1/Level 2 GCSE in Computer Science is based on and mapped against the Computer Science curriculum for schools produced by the CAS Working Group. See *Appendix 5* for further details.

The qualification is also benchmarked against the curricula of high-performing jurisdictions in other parts of the world, notably India, Israel and Singapore. See 'International Comparisons' a briefing note produced by CAS (November 2011).

Pearson has consulted widely with members of the computing community – employers, higher education institutions and teachers – to ensure that the content of this GCSE is relevant, fit for purpose and supports progression into higher education and ultimately into employment.

Assessment

While we recognise that programming is an important element of computer science, we feel strongly that the underlying principles of logic, decomposition, algorithms, data representation, communication etc. are even more fundamental and durable. This is reflected in the assessment model used for this qualification.

The written paper, 'Principles of Computer Science', is a rigorous, intellectually challenging examination with a weighting of 75% that requires a high level of computational thinking.

Practical programming skills are assessed in the controlled assessment, which has a weighting of 25%.

This is similar to the approach taken by high-performing jurisdictions in other parts of the world, such as India, Israel and Singapore.

Qualification objectives

The aims of the Pearson Edexcel Level 1/Level 2 GCSE in Computer Science are to enable learners to:

- develop knowledge and understanding of the fundamental principles and concepts of computer science
- develop and apply computational thinking skills to analyse problems and design solutions across a range of contexts
- gain practical experience of designing, writing, and testing computer programs that accomplish specific goals
- develop the ability to reason, explain and evaluate computing solutions
- develop awareness of current and emerging trends in computing technologies
- develop awareness of the impact of computing on individuals, society and the environment, including ethical, legal and ownership issues
- communicate computer science concepts and explain computational solutions clearly and concisely using appropriate terminology.

Contents

Specification at a glance

The Pearson Edexcel Level 1/Level 2 GCSE in Computer Science is a linear qualification. It has two assessment components:

- paper-based assessment: Principles of Computer Science
- controlled assessment: Practical Programming.

Paper-based assessment: Principles of Computer Science	***Unit code: 1CP0/01**
Externally assessed Availability: June First assessment: June 2015	**75% of the total GCSE**

Overview of content

- Understanding of what algorithms are, what they are used for and how they work; ability to interpret, amend and create algorithms.
- Understanding of binary representation, data representation, data storage and compression, encryption and databases; ability to use SQL to insert, amend and extract data stored in a structured database.
- Understanding of components of computer systems; ability to construct truth tables, produce logic statements and read and interpret fragments of assembly code.
- Understanding of computer networks, the internet and the world wide web; ability to use HTML and CSS to construct web pages.
- Awareness of emerging trends in computing technologies, the impact of computing on individuals, society and the environment, including ethical, legal and ownership issues.

Overview of assessment

The assessment is paper based.

The assessment is 120 minutes.

The assessment consists of 5 questions.

The assessment consists of 90 marks.

Each question is set in a context, draws on topics from across the specification and is broken down into a number of parts.

The assessment includes extended writing that assesses Quality of Written Communication.

Controlled assessment: Practical Programming	***Unit codes: 1CP0/2A, 1CP0/2B, 1CP0/2C**
Internally assessed Availability: June First assessment: June 2015	**25% of the total GCSE**

Overview of content

This is a practical 'making task' that enables students to demonstrate their computational techniques using a programming language. Students will:

- decompose problems into sub-problems
- create original algorithms or work with algorithms produced by others
- design, write, test, and evaluate programs.

Overview of assessment

Controlled assessment tasks will be released each January.

The assessment will be carried out at a computer under controlled conditions.

The controlled assessment may take place over multiple sessions up to a combined duration of 15 hours.

The assessment consists of 3 tasks.

The assessment consists of 50 marks.

Students must select one programming language from the following:

- Python (unit code 1CP0/2A)
- Java (unit code 1CP0/2B)
- C-derived language (unit code 1CP0/2C).

The assessment includes extended writing that assesses Quality of Written Communication.

* See *Appendix 3* for a description of these codes and all other codes relevant to this qualification.

Qualification content

Topic 1: Problem solving

Students are expected to develop a set of computational thinking skills that enable them to understand how computer systems work, and design, implement and analyse algorithms for solving problems.

Students should be given repeated opportunities to tackle computational problems of various sorts, including some substantial problem-solving tasks.

Subject content	Students should:
1.1 Algorithms	1.1.1 Understand what an algorithm is, what algorithms are used for and be able to interpret algorithms [flowcharts, pseudocode, structured English, written descriptions, program code]*
	1.1.2 Be able to create an algorithm to solve a particular problem, making use of programming constructs [sequence, selection, repetition] and using an appropriate notation [flowchart, written description, program code]
	1.1.3 Be able to describe the purpose of a given algorithm and explain how a simple algorithm works
	1.1.4 Be able to identify the correct output of an algorithm for a given set of data
	1.1.5 Be able to identify and correct errors in algorithms
	1.1.6 Be able to code an algorithm into a high-level language
	1.1.7 Understand how the choice of algorithm is influenced by the data structure and data values that need to be manipulated
	1.1.8 Understand how standard algorithms [quick sort, bubble sort, selection sort, linear search, binary search, breadth first search, depth first search, maximum/minimum, mean, count] work
	1.1.9 Understand factors that affect the efficiency of an algorithm
1.2 Decomposition	1.2.1 Be able to analyse a problem, investigate requirements [inputs, outputs, processing, initialisation] and design solutions
	1.2.2 Be able to decompose a problem into smaller sub-problems

* See *Appendix 2* for key definitions related to this content.

Topic 2: Programming

Learning to program is a core component of a computer science course. Students should be competent at reading and writing programs and be able to reason about code. They must be able to apply their skills to solve real problems and produce robust programs.

Students should be given repeated opportunities to develop and practise their programming skills.

Subject content	Students should:
2.1 Develop code	2.1.1 Be able to write programs in a high-level programming language
	2.1.2 Understand the benefit of producing programs that are easy to read, and be able to use techniques [comments, descriptive variable names, indentation] to improve readability and to explain how the code works
	2.1.3 Be able to differentiate between types of error in programs [logic, syntax, runtime]
	2.1.4 Be able to design and use test plans and test data
	2.1.5 Be able to interpret error messages and identify, locate and fix errors in a program
	2.1.6 Be able to identify what value a variable will hold at a given point in a program [trace table]
	2.1.7 Be able to make effective use of tools offered in an integrated development environment [watcher, break points, single-step, step-throughs]
	2.1.8 Be able to evaluate the strengths and weaknesses of a program and suggest improvements
	2.1.9 Be able to work safely, respectfully, responsibly and securely when using computers
2.2 Constructs	2.2.1 Be able to identify the structural components of a program [variable and type declarations, initialisations, command sequences, conditionals, repetition, data structures, subprograms]
	2.2.2 Be able to use sequencing, selection and repetition constructs in their programs
2.3 Data types and structures	2.3.1 Understand the need for and be able to select and use data types [integer, real, Boolean, char]
	2.3.2 Understand the need for and be able to select and use data structures [one-dimensional arrays, two-dimensional arrays]
	2.3.3 Understand the need for and be able to manipulate strings
	2.3.4 Understand the need for and be able to use variables and constants
	2.3.5 Understand the need for and be able to use global and local variables

Subject content	Students should:
2.4 Input/output	2.4.1 Be able to write code that accepts and responds appropriately to user input
	2.4.2 Understand the need for and be able to implement validation
	2.4.3 Be able to write code that outputs information to a screen and understand and use Cartesian x/y coordinates
	2.4.4 Be able to design and code a user interface [textual, graphical]
	2.4.5 Be able to write code that opens/closes, reads/writes, deletes, inserts, appends from/to a file
2.5 Operators	2.5.1 Understand the purpose of and be able to use arithmetic operators [plus, minus, divide, multiply, modulus, integer division]
	2.5.2 Understand the purpose of and be able to use relational operators [equal to, less than, greater than, not equal to, less than or equal to, greater than or equal to]
	2.5.3 Understand the purpose of and be able to use Boolean operators [AND, OR, NOT]
2.6 Subprograms	2.6.1 Understand the benefits of using subprograms and be able to write code that uses user-written and pre-existing [built-in, library] subprograms
	2.6.2 Understand the concept of passing data into and out of subprograms [procedures, functions, return values]
	2.6.3 Be able to create subprograms that perform generalisation

Topic 3: Data

Computers are able to store and manipulate large quantities of data. They use binary to represent different types of data.

Students are expected to learn how different types of data are represented in a computer. They should be given the opportunity to gain practical experience of using SQL to handle data stored in a structured database.

Subject content	Students should:
3.1 Binary	3.1.1 Understand that computers use binary to represent data and instructions
	3.1.2 Understand how computers represent and manipulate numbers [unsigned integers, signed integers (sign and magnitude, Two's complement) real numbers]
	3.1.3 Be able to convert between binary and denary whole numbers (0-255) and vice versa
	3.1.4 Be able to perform binary arithmetic [add, subtract, multiply] and understand the concept of overflow
	3.1.5 Understand why hexadecimal notation is used and be able to convert between hexadecimal and binary and vice versa
3.2 Data representation	3.2.1 Understand how computers encode characters [ASCII, Unicode]
	3.2.2 Understand how bitmap images are represented in binary [pixels, resolution, colour depth]
	3.2.3 Understand how analogue data [sound, temperature, light intensity] is represented in binary
	3.2.4 Understand the limitations of binary representation of data [quantisation, sampling frequency] and how bit length constrains the range of values that can be represented
3.3 Data storage and Compression	3.3.1 Understand and be able to convert between the terms 'bit, nibble, byte, kilobyte (KB), megabyte (MB), gigabyte (GB), terabyte (TB)'
	3.3.2 Understand the need for data compression and methods of compressing data [lossless, lossy] and that JPEG and MP3 are examples of lossy algorithms
	3.3.3 Understand how a lossless, run-length encoding [RLE] algorithm works
	3.3.4 Understand that file storage is measured in bytes and that data transmission is measured in bits per seconds, and be able to calculate the time required to transmit a file and storage requirements for files
3.4 Encryption	3.4.1 Understand the need for data encryption
	3.4.2 Understand how a Caesar cipher algorithm works

Subject content	Students should:
3.5 Databases	3.5.1 Understand the characteristics of structured and unstructured data
	3.5.2 Understand that data can be decomposed and organised in a structured database [tables, records, fields, relationships, keys]
	3.5.3 Understand the need for and be able to use SQL statements *

* See *Appendix 2* for key definitions related to this content.

Topic 4: Computers

Students must be familiar with the hardware and software components that make up a computer system and recognise that computers come in all shapes and sizes from embedded microprocessors to distributed clouds.

Students should be given the opportunity to gain practical experience of interpreting instructions written in assembly language.

Subject content	Students should:
4.1 Machines and computational models	4.1.1 Understand the concept of a computer as a hardware machine or as a virtual machine
	4.1.2 Understand that there is a range of computational models [sequential, parallel, multi-agent]
	4.1.3 Understand the input-process-output model
4.2 Hardware	4.2.1 Understand the function of hardware components of a computer system [processor (CPU), memory, secondary storage, input devices, output devices] and how they work together
	4.2.2 Understand the concept of a stored program and the role of components of the processor [control unit (CU), arithmetic/logic unit (ALU), registers, clock, address bus, data bus] in the fetch-decode-execute cycle
	4.2.3 Understand the function of assembly code and be able to interpret a block of assembly code using a given set of commands*
	4.2.4 Understand how data is stored on physical devices [magnetic, optical, solid state]
	4.2.5 Understand how microcontrollers can be programmed to control actuators and take input from sensors
4.3 Logic	4.3.1 Be able to construct truth tables for a given logic statement [AND, OR, NOT]
	4.3.2 Be able to produce logic statements for a given problem
4.4 Software	4.4.1 Understand what an operating system is and the functions of an operating system [file management, input/output, resource allocation, process management, network management, user management]
	4.4.2 Understand that application software such as a web browser, word processor, spreadsheet or apps are computer programs
	4.4.3 Understand how software can be used to simulate and model aspects of the real world and be able to create software models
4.5 Programming languages	4.5.1 Understand what is meant by high-level and low-level programming languages and assess their suitability for a particular task
	4.5.2 Understand what is meant by a compiler and an interpreter

* See *Appendix 2* for key definitions related to this content.

Topic 5: Communication and the internet

Computer networks and the internet are now ubiquitous. Many computer applications in use today would not be possible without networks. Students should understand the key principles behind the organisation and of computer networks. Ideally, they should be able to experiment by setting up a simple network.

Students should be given the opportunity to gain practical experience of creating web pages.

Subject content	**Students should:**
5.1 Networks	5.1.1 Understand why computers are connected in a network
	5.1.2 Understand the different types of networks [LAN, WAN, PAN, VPN]
	5.1.3 Understand the network media [copper cable, fibre optic cable, wireless]
	5.1.4 Understand that network data speeds are measured in bits per second [Mbps, Gbps]
	5.1.5 Understand the role of and need for network protocols
	5.1.6 Understand that data can be transmitted over networks using packets [TCP/IP]
	5.1.7 Understand the need to detect and correct errors in data transmission [check sums]
	5.1.8 Understand the concept of and need for network addressing and host names [MAC addresses]
	5.1.9 Understand characteristics of network topologies [bus, ring, star, mesh]
5.2 The internet and the world wide web	5.2.1 Understand what is meant by the internet and how the internet is structured [IP addressing, routers, connecting backbone, domain names]
	5.2.2 Understand what is meant by the world wide web (WWW) and components of the WWW [web server URLs, ISP, HTTP, HTTPS, HTML]
	5.2.3 Be able to use HTML and CSS to construct web pages [formatting, links, images, media, layout, styles, lists]
	5.2.4 Understand the client-server model, the difference between client-side and server-side processing and the role of cookies

Topic 6: The bigger picture

Students should be aware of emerging trends in computing technology and recognise that computing has an impact on nearly every aspect of the world in which they live.

Subject content	Students should:
6.1 Emerging trends, issues and impact	6.1.1 Be aware of current and emerging trends in computing technology [quantum computing, DNA computing, artificial intelligence (AI), nano technology]
	6.1.2 Be aware of the impact of computing on individuals, society and the environment
	6.1.3 Be aware of ethical and legal issues arising from the use of computers
	6.1.4 Be aware of ownership issues relating to computing [intellectual property, patents, licensing, open source and proprietary software]

Assessment

Assessment summary

Summary of table of assessment

Paper-based assessment: Principles of Computer Science	***Unit code: 1CP0/01**
Externally assessed Availability: June First assessment: June 2015	**75% of the total GCSE**

Overview of content

- Understanding of what algorithms are, what they are used for and how they work; ability to interpret, amend and create algorithms.
- Understanding of binary representation, data representation, data storage and compression, encryption and databases; ability to use SQL to insert, amend and extract data stored in a structured database.
- Understanding of components of computer systems; ability to construct truth tables, produce logic statements and read and interpret fragments of assembly code.
- Understanding of computer networks, the internet and the world wide web; ability to use HTML and CSS to construct web pages.
- Awareness of emerging trends in computing technologies, the impact of computing on individuals, society and the environment, including ethical, legal and ownership issues.

Overview of assessment

The assessment is paper based.

The assessment is 120 minutes.

The assessment consists of 5 questions.

The assessment consists of 90 marks.

Each question is set in a context, draws on topics from across the specification and is broken down into a number of parts.

The assessment includes extended writing that assesses Quality of Written Communication.

Controlled assessment: Practical Programming	***Unit codes: 1CP0/2A, 1CP0/2B, 1CP0/2C**
Internally assessed Availability: June First assessment: June 2015	**25% of the total GCSE**

Overview of content

This is a practical 'making task' that enables students to demonstrate their computational techniques using a programming language. Students will:

- decompose problems into sub-problems
- create original algorithms or work with algorithms produced by others
- design, write, test and evaluate programs.

Overview of assessment

Controlled assessment tasks will be released each January.

The assessment will be carried out at a computer under controlled conditions.

The controlled assessment may take place over multiple sessions up to a combined duration of 15 hours.

The assessment consists of 3 tasks.

The assessment consists of 50 marks.

Students must select one programming language from the following:

- Python (unit code 1CP0/2A)
- Java (unit code 1CP0/2B)
- C-derived language (unit code 1CP0/2C).

The assessment includes extended writing that assesses Quality of Written Communication.

*See *Appendix 3* for a description of these codes and all other codes relevant to this qualification.

Assessment Objectives and weightings

Students must:			% in GCSE
AO1	AO1	Recall and demonstrate knowledge and understanding of computer science	32-42%
AO2	AO2	Apply knowledge and understanding, solve problems and develop solutions	40-50%
AO3	AO3	Evaluate, make reasoned judgements and draw conclusions	15-25%
		TOTAL	100%

Relationship of Assessment Objectives to papers

Unit	Assessment Objective			
	AO1	**AO2**	**AO3**	**Total for AO1, AO2 and AO3**
Paper 1: **Principles of Computer Science**	27%-32%	30%-35%	10%-15%	75%
Controlled assessment: **Practical Programming**	5%-10%	10%-15%	5%-10%	25%
Total for GCSE	32-42%	40-50%	15-25%	100%

Entering your students for assessment

Student entry

Details of how to enter students for the examinations for this qualification can be found in our Information Manual, a copy is sent to all examinations officers. The information can also be found on our website: www.edexcel.com

Students are required to sit all of their examinations at the end of the course. The controlled assessment task will be made available on our website for teachers to download in January of the terminal year. Work must be submitted for moderation at the end of the course. Centres must ensure that controlled assessment tasks submitted are valid for the series in which they are submitted.

Forbidden combinations and classification code

Centres should be aware that students who enter for more than one GCSE qualification with the same classification code will have only one grade (the highest) counted for the purpose of the School and College Performance Tables (please see *Appendix 3*).

Students should be advised that, if they take two qualifications with the same classification code, schools and colleges are very likely to take the view that they have achieved only one of the two GCSEs. The same view may be taken if students take two GCSE qualifications that have different classification codes but have significant overlap of content. Students who have any doubts about their subject combinations should check with the institution to which they wish to progress before embarking on their programmes.

Access arrangements and special requirements

Pearson's policy on access arrangements and special considerations for GCE, GCSE, and Entry Level is designed to ensure equal access to qualifications for all students (in compliance with the Equality Act 2010) without compromising the assessment of skills, knowledge, understanding or competence.

Please see the Joint Council for Qualifications (JCQ) website (www.jcq.org.uk) for its policy on access arrangements, reasonable adjustments and special considerations.

Please see our website (www.edexcel.com) for:

- the forms to submit for requests for access arrangements and special considerations
- dates to submit the forms.

Requests for access arrangements and special considerations must be addressed to:

Special Requirements
Pearson Education Limited
One90 High Holborn
London WC1V 7BH

Equality Act 2010

Please see our website (www.edexcel.com) for information on the Equality Act 2010.

Assessing your students

The first assessment opportunity for this qualification will take place in the June 2015 series and in each following June series for the lifetime of the specification.

Your student assessment opportunities

Unit	June 2015	June 2016
Paper 1	✓	✓
Controlled assessment	✓	✓

All assessments must be taken at the end of the course.

Awarding and reporting

The grading, awarding and certification of this qualification will comply with the requirements of the current GCSE/GCE Code of Practice, which is published by the Office of Qualifications and Examinations Regulation (Ofqual).

The two assessed components of GCSE Computer Science are scaled as follows to create a total subject mark:

Assessed component	Percentage weighting	Raw mark × scaling factor	Subject mark
Paper 1	75%	90 × 1.6	150
Controlled assessment	25%	50 × 1	50
		Total subject mark	200

The GCSE qualification will be graded and certificated on an eight-grade scale from A* to G using the total subject mark. Individual components are not graded.

The first certification opportunity for the Pearson Edexcel Level 1/ Level 2 GCSE in Computer Science will be 2015.

Students whose level of achievement is below the minimum judged by Pearson to be of sufficient standard to be recorded on a certificate will receive an unclassified U result.

Stretch and challenge

Students can be stretched and challenged in all units through the use of different assessment strategies, for example:

- using a variety of stems in questions, for example analyse, evaluate, discuss, compare
- ensuring connectivity between sections of questions
- a requirement for extended writing
- using a wider range of question types to address different skills, for example open-ended questions, case studies.

Malpractice and plagiarism

For up-to-date advice on malpractice and plagiarism, please refer to the latest Joint Council for Qualifications (JCQ) Instructions for Conducting Coursework document. This document is available on the JCQ website: www.jcq.org.uk.

For additional information on malpractice, please refer to the latest Joint Council for Qualifications (JCQ) *Suspected Malpractice in Examinations and Assessments: Policies and Procedures* document, available on the JCQ website.

Student recruitment

Pearson's access policy concerning recruitment to our qualifications is that:

- they must be available to anyone who is capable of reaching the required standard
- they must be free from barriers that restrict access and progression
- equal opportunities exist for all students.

Prior knowledge and skills

There are no prior learning requirements for this qualification.

Progression

Students can progress from this qualification to a GCE in Computing or to Pearson BTEC National Qualifications in ICT.

Grade descriptions

A	Students recall, select and communicate precise knowledge and detailed understanding of computer science. They demonstrate a comprehensive understanding of how computers and computer systems work. They use technical knowledge, terminology and conventions appropriately and consistently. They apply appropriate computational thinking skills to systematically analyse, model and solve problems. They use a wide range of computational concepts and practices appropriately to develop effective solutions. They critically evaluate and draw detailed, informed conclusions about the way they and others use computer science to solve problems, making reasoned judgements about the effectiveness of solutions and suggesting realistic improvements.
C	Students recall, select and communicate a secure knowledge and understanding of computer science. They demonstrate understanding of how computers and computer systems work. They use technical knowledge, terminology and conventions appropriately. They apply some appropriate computational thinking skills to analyse, model and solve problems. They use a range of computational concepts and practices appropriately to develop solutions. They evaluate and draw conclusions about the way they and others use computer science to solve problems, drawing conclusions about the effectiveness of solutions and suggesting some realistic improvements.
F	Students recall, select and communicate a basic knowledge and understanding of aspects of computer science. They have a limited understanding of how computers and computer systems work. They use limited technical knowledge, terminology and conventions. They apply limited computational thinking skills to analyse, model and solve problems. They use a limited range of computational concepts and practices to develop basic solutions. They comment on the way they and others use computer science to solve problems, drawing elementary conclusions about the effectiveness of solutions and suggesting some simple improvements.

Controlled assessment

Controlled assessment summary

Overview

The purpose of the controlled assessment is to test student skills in responding to computer science problems. In the controlled assessment, evidence of skills will be demonstrated through responses to **three** tasks set in a context:

- students may spend a maximum of 15 hours working on the controlled assessment tasks
- the controlled assessment brief specifies the recommended duration for each task
- levels of control:
  - high task setting
  - medium task taking
  - medium task marking.

Students must select one programming language from the following:

- Python (unit code 1CP0/2A)
- Java (unit code 1CP0/2B)
- C-derived language (unit code 1CP0/2C).

Programming languages:

If candidates use software development kits, they must indicate which code they have written.

Levels of control

Task setting

High level of control

A high level of control means that Pearson will set tasks for students to complete. The controlled assessment task will be made available on our website for teachers to download in January of the terminal year.

The format of the tasks will be similar each year but the context will change. Teachers must ensure that students are completing the correct controlled assessment task for their terminal year. The front sheet shows the dates for which it is valid.

Task taking

Medium level of control

Authenticity control

Students must work alone to develop a response to the task. Students and teachers must sign the Controlled Assessment Authentication Statement (*Appendix 4*).

Collaboration control

Students must be supervised and may not work with others to develop a response to the task.

Feedback control

Teachers may help students to understand rubrics, assessment criteria and controlled conditions. Teachers may not provide solutions to students. Any additional feedback must be recorded on the Controlled Assessment Authentication Statement (*Appendix 4*).

Resources control

Every student must each have equal access to IT resources. Internet access is not allowed. Where students are taking the controlled assessment over a number of sessions, at the end of each session their work must be saved and kept securely. Students must not be able to access their work outside the controlled assessment setting.

In situations where computer workstations are situated close together invigilators must ensure that students are working independently.

Time control

A maximum of 15 hours' total duration is permitted for students to complete the assessment. Suggested times will be indicated for each task on the Controlled Assessment Task. This may be divided into shorter sessions. Where students are taking the controlled assessment over a number of sessions, at the end of each session their work must be saved and kept securely.

Task marking

Medium level of control

Teachers should mark the controlled assessment using the assessment criteria on the following pages. There is no requirement to annotate students' work, although it is necessary to write full justification comments on the Controlled Assessment Authentication Statement (*Appendix 4*).

Students may provide solutions to the tasks in one of the programming languages specified by Pearson. Assessment criteria are applicable to all these languages.

Quality of Written Communication

Quality of Written Communication (QWC) will be assessed in the task. It will assess students on their ability to:

- present relevant information in a form that suits its purpose
- ensure that text is legible and that spelling, punctuation and grammar are accurate, so that meaning is clear
- use suitable structure and style of writing
- use specialist vocabulary when appropriate.

Presentation of work

Students must present their work for the controlled assessment electronically. Student files should be identified by student name and task and presented in a single folder.

Students will be expected to present content in a format appropriate for viewing at a resolution of 1024 x 768 pixels.

Marking, standardisation and moderation

The controlled assessment is marked by centre staff. Where marking for this specification has been carried out by more than one teacher in a centre, there must be a process of internal standardisation carried out to ensure that there is a consistent application of the criteria laid down in the marking grids, across all the units.

Marks awarded by the centre will be subject to external moderation by Pearson. This is to ensure consistency with national standards. Following the submission of marks, Pearson will notify centres of the students whose responses have been selected for moderation. This sample will take cohort size into account.

Work must be submitted in an approved digital format.

If the moderation indicates that centre assessment does not reflect national standards, an adjustment will be made to students' final marks to compensate for this. Please refer to the JCQ Instructions for conducting Controlled Assessments (GCSE qualifications) on the JCQ website: www.jcq.org.uk for further information. The controlled assessment in this qualification will comply with these instructions.

Security and backups

It is the responsibility of the centre to keep secure the work that students have submitted for assessment. Centres are strongly advised to utilise firewall protection and virus checking software and to employ an effective backup strategy, so that an up-to-date archive of students' evidence is maintained.

Centres are advised to archive completed, assessed work so as to free up work space for work in progress.

Language of assessment

Assessment of these units will be available in English. All student work must be in English.

Further information

For more information on annotation, authentication, mark submission and moderation procedures, please refer to *Moderation of Controlled Assessments: Guidance for Centres for Pearson Edexcel Level 1/Level 2 GCSE in Computer Science*, available on our website.

For up-to-date advice on teacher involvement, please refer to the Joint Council for Qualifications (JCQ) *Instructions for conducting coursework (GCSE qualifications)* document on the JCQ website: www.jcq.org.uk.

For up-to-date advice on malpractice and plagiarism, please refer to the Joint Council for Qualifications (JCQ) *Suspected Malpractice in Examinations and Assessments* document on the JCQ website (www.jcq.org.uk).

Controlled assessment criteria

Teachers must mark students' work using the assessment criteria specified below.

Task number 1		**Programming: Targeting AO1/AO2**	**8 marks**
Level	**Mark**	**Descriptor**	
	0	No rewardable material.	
1	1–2	Program only partially addresses the requirements. Some of the programming constructs selected are not appropriate. The program runs with some errors. Variable names, layout and structure do not aid readability. Comments do not explain how the program works.	
2	3–5	Program addresses some of the requirements. The programming constructs selected are mostly appropriate. The program runs without major errors. Variable names, layout and structure make parts of the program easy to read. Comments partially explain how the program works.	
3	6–8	Program fully addresses all of the requirements. The programming constructs selected are appropriate and together produce an efficient solution. The program runs without errors. Variable names, layout and structure make the whole program easy to read. Comments fully explain how the program works.	

Task number 2a		Programming: Targeting AO1/AO2	12 marks
Level	**Mark**	**Descriptor**	
	0	No rewardable material.	
1	1–4	Program only partially addresses the requirements. The program is poorly designed with little or no attempt at decomposition. Some of the programming constructs selected are not appropriate. The program runs with some errors. There is limited evidence of testing. Variable names, layout and structure do not aid readability. Comments do not explain how the program works.	
2	5–8	Program addresses most of the requirements. Some aspects of the program are well designed with partial decomposition into subprograms. The programming constructs selected are mostly appropriate. The program runs without major errors. There is evidence that testing of most aspects of the program has been planned and carried out. Variable names, layout and structure make most of the program easy to read. Comments partially explain how the program works.	
3	9–12	Program fully addresses all of the requirements. The whole program is well designed and fully decomposed into subprograms. The programming constructs selected are appropriate and together produce an efficient solution. The program runs without errors. There is evidence that thorough testing of the whole program has been planned and carried out. Variable names, layout and structure make the whole program easy to read. Comments fully explain how the program works.	

Task number 2b		Evaluation: Targeting AO3	6 marks
Level	**Mark**	**Descriptor**	
	0	No rewardable material.	
1 QWC i-ii-iii	1–2	Some attempt to describe how the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates no understanding of the computational thinking involved in solving the problem. Specialist technical terms have not been used appropriately and response lacks clarity and organisation. Spelling, punctuation and the rules of grammar are used with little accuracy and with frequent errors throughout.	
2 QWC i-ii-iii	3–4	Evaluative comments consider how well the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates some understanding of the computational thinking involved in solving the problem. Some specialist technical terms are used appropriately and the response shows focus and organisation. Spelling and punctuation and the rules of grammar are mostly applied accurately with some errors throughout.	
3 QWC i-ii-iii	5–6	A thorough evaluation that critically reviews how successfully the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates a thorough understanding of the computational thinking involved in solving the problem. Specialist technical terms are used appropriately and extensively and the response shows good focus and organisation. Spelling, punctuation and the rules of grammar are used consistently and accurately throughout.	

Task number 3a		Programming: Targeting AO1/AO2	15 marks
Level	**Mark**	**Descriptor**	
	0	No rewardable material.	
1	1–5	Program only partially addresses the requirements. The program is poorly designed with little or no attempt at decomposition. Some of the programming constructs selected are not appropriate. The program runs with some errors. There is limited evidence of testing. Variable names, layout and structure do not aid readability. Comments do not explain how the program works.	
2	6–10	Program addresses most of the requirements. Some aspects of the program are well designed with partial decomposition into subprograms. The programming constructs selected are mostly appropriate. The program runs without major errors. There is evidence that testing of most aspects of the program has been planned and carried out. Variable names, layout and structure make most of the program easy to read. Comments partially explain how the program works.	
3	11–15	Program fully addresses all of the requirements. The whole program is well designed and fully decomposed into subprograms. The programming constructs selected are appropriate and together produce an efficient solution. The program runs without errors. There is evidence that thorough testing of the whole program has been planned and carried out. Variable names, layout and structure make the whole program easy to read. Comments fully explain how the program works.	

Task number 3b		Evaluation: Targeting AO3	9 marks
Level	**Mark**	**Descriptor**	
	0	No rewardable material.	
1 QWC i-ii-iii	1–3	Some attempt to describe how the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates no understanding of the computational thinking involved in solving the problem. Specialist technical terms have not been used appropriately and response lacks clarity and organisation. Spelling, punctuation and the rules of grammar are used with little accuracy and with frequent errors throughout.	
2 QWC i-ii-iii	4–6	Evaluative comments consider how well the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates some understanding of the computational thinking involved in solving the problem. Some specialist technical terms are used appropriately and the response shows focus and organisation. Spelling and punctuation and the rules of grammar are mostly applied accurately with some errors throughout.	
3 QWC i-ii-iii	7–9	A thorough evaluation that critically reviews how successfully the program meets the specified requirements. Explanation of how a specified aspect of the program works demonstrates a thorough understanding of the computational thinking involved in solving the problem. Specialist technical terms are used appropriately and extensively and the response shows good focus and organisation. Spelling, punctuation and the rules of grammar are used consistently and accurately throughout.	

Resources, support and training

Pearson resources

Pearson aims to provide and endorse the most comprehensive support for our qualifications.

Pearson publications

You can order further copies of this Specification and Sample Assessment Materials (SAMs) documents via our website www.edexcel.com

Endorsed resources

Pearson also endorses some additional materials written to support this qualification. Any resources bearing the Pearson Edexcel logo have been through a quality assurance process to ensure complete and accurate support for the specification. For up-to-date information about endorsed resources, please visit www.edexcel.com/endorsed

Please note that while resources are checked at the time of publication, materials may be withdrawn from circulation and website locations may change.

Pearson support services

Pearson has a wide range of support services to help you implement this qualification successfully.

ResultsPlus – ResultsPlus is an application launched by Pearson to help subject teachers, senior management teams, and students by providing detailed analysis of examination performance. Reports that compare performance between subjects, classes, your centre and similar centres can be generated in 'one-click'. Skills maps that show performance according to the specification topic being tested are available for some subjects. For further information about which subjects will be analysed through ResultsPlus, and for information on how to access and use the service, please visit www.edexcel.com/resultsplus

Ask the Expert – to make it easier for our teachers to ask us subject specific questions we have provided the **Ask the Expert** Service. This easy-to-use web query form will allow you to ask any question about the delivery or teaching of Pearson qualifications. You'll get a personal response, from one of our administrative or teaching experts, sent to the email address you provide. You can access this service at www.edexcel.com/ask

Support for students

Learning flourishes when students take an active interest in their education; when they have all the information they need to make the right decisions about their futures. With the help of feedback from students and their teachers, we've developed a website for students that will help them:

- understand subject specifications
- access past papers and mark schemes
- learn about other students' experiences at university, on their travels and when entering the workplace.

We're committed to regularly updating and improving our online services for students. The most valuable service we can provide is helping schools and colleges unlock the potential of their students. www.edexcel.com/students

Training

A programme of professional development and training courses, covering various aspects of the specification and examination, will be arranged by Pearson each year on a regional basis. Full details can be obtained from:

Training from Edexcel
Pearson Education Limited
One90 High Holborn
London
WC1V 7BH

Telephone: 0844 576 0027
Email: trainingbookings@pearson.com
Website: www.edexcel.com

Appendices

Appendix 1 Wider curriculum

Signposting

Issue	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6
Moral						✓
Ethical		✓				✓
Social						✓
Citizenship		✓				✓
Environmental	✓			✓		✓

Development suggestions

Issue	Topic	Opportunities for development or internal assessment
Moral	6	• considering the impact of computing on individuals and society. • being aware of issues relating to ownership of hardware and software.
Ethical	2 6	• being aware of the ethical and legal issues arising from the use of computers. • being aware of issues relating to ownership of hardware and software. • recognising the importance of working respectfully and responsibly.
Social	6	• considering the impact of computing on individuals and society. • recognising the importance of working respectfully and responsibly.
Citizenship	2 6	• recognising the importance of incorporating ethical and legal principles into own practice when creating computer programs. • recognising the importance of working respectfully and responsibly.
Environmental	1 4 6	• considering ways of minimising/mitigating the environmental impact of computers.

Appendix 2 Key definitions

Pseudocode Command Set

Questions in the written examination that involve code will use this pseudocode for clarity and consistency. However, students may answer questions using any valid method.

Data types

INTEGER
REAL
BOOLEAN
CHARACTER

Data structures

ARRAY
STRING

Identifiers

Identifiers are sequences of letters, digits and '_', starting with a letter,
e.g. MyValue, My_Value, Counter2

Functions

LENGTH
RANDOM

Variables and arrays		
Syntax	**Description**	**Example**
SET Variable TO <value>	Assigns a value to a variable.	SET Counter TO 0 SET MyString TO 'Hello world'
SET Variable TO <expression>	Computes the value of an expression and assigns to a variable.	SET Sum TO Score + 10 SET Size to LENGTH(Word)
SET Array[index] TO <value>	Assigns a value to an element of a one-dimensional array, where index is an integer value starting at 0.	SET ArrayClass[1] TO 'Ann' SET ArrayMarks[3]TO 56
SET Array TO [<value>, …]	Initialises a one-dimensional array with a set of values .	SET ArrayValues TO [1, 2, 3, 4, 5]
SET Array [RowIndex, ColumnIndex] TO <value>	Assigns a value to an element of a two dimensional array, where RowIndex and ColumnIndex are integer values starting at O.	SET ArrayClassMarks[2,4] TO 92

Selection		
Syntax	**Description**	**Example**

IF <expression> THEN
<command>
END IF
	If <expression> is true then command is executed.	IF Answer = 10 THEN
		  SET Score TO Score + 1
		END IF

IF <expression> THEN
<command>
ELSE
<command>
END IF
	If <expression> is true then first <command> is executed, otherwise second <command> is executed.	IF Answer = 'correct' THEN
		  SEND 'Well done' TO DISPLAY
		ELSE
		  SEND 'Try again' TO DISPLAY
		END IF

Repetition		
Syntax	**Description**	**Example**

WHILE <condition> DO
<command>
END WHILE
	Pre-conditioned loop. Executes <command> whilst <condition> is true.	WHILE Flag = 0 DO
		SEND 'All well' TO DISPLAY
		END WHILE

REPEAT <command> UNTIL <expression>	Post-conditioned loop. Executes <command> until <condition> is true. The loop must execute at least once.	REPEAT SET Go TO Go + 1 UNTIL Go = 10
REPEAT <expression> TIMES <command> END REPEAT	Count controlled loop. The number of times <command> is executed is determined by the expression.	REPEAT 100-Number TIMES SEND '* ' TO DISPLAY END REPEAT

FOR <id> FROM <expression> TO <expression> DO
<command>
END FOR
	Count controlled loop. Executes <command> a fixed number of times.	FOR Index FROM 1 TO 10 DO
		SEND ArrayNumbers[Index] TO DISPLAY
		END FOR

FOR <id> FROM <expression> TO <expression> STEP <expression> DO
<command>
END FOR
	Count controlled loop using a step.	FOR Index FROM 1 TO 500 STEP 25 DO
		SOUND Buzzer(10)
		END FOR

FOR EACH <id> FROM <expression> DO <command> END FOREACH	Count controlled loop. Executes for each element of an array.	SET WordsArray TO ['The', 'Sky', 'is', 'grey' SET Sentence to '' FOR EACH Word FROM WordsArray DO SET Sentence TO Sentence & Word & ' ' END FOREACH

Input/output		
Syntax	**Description**	**Example**
SEND <expression> TO DISPLAY	Sends output to the screen.	SEND 'Have a good day.' TO DISPLAY
RECEIVE <identifier> FROM (type) <device>	Reads input of specified type.	RECEIVE Name FROM (STRING) KEYBOARD RECEIVE LengthOfJourney FROM (INTEGER) CARD_READER RECEIVE YesNo FROM (CHARACTER) CARD_READER

File handling		
Syntax	**Description**	**Example**
READ <File> <record>	Reads in a record from a <file> and assigns to a <variable>. Each READ statement reads a record from the file.	READ MyFile.doc Record
WRITE <File> <record>	Writes a record to a file. Each WRITE statement writes a record to the file.	WRITE MyFile.doc Answer1, Answer2, 'xyz 01'

Subprograms		
Syntax	**Description**	**Example**
PROCEDURE <id> (<parameter>, …) BEGIN PROCEDURE <command> END PROCEDURE	Defines a procedure.	PROCEDURE CalculateAverage (Mark1, Mark2, Mark3) BEGIN PROCEDURE SET Avg to (Mark1 + Mark2 + Mark3)/3 END PROCEDURE

FUNCTION <id> (<parameter>, …)
BEGIN FUNCTION
    <command>
    RETURN <expression>
END FUNCTION
	Defines a function.	FUNCTION AddMarks (Mark1, Mark2, Mark3)
		BEGIN FUNCTION
		    SET Total to (Mark1 + Mark2 + Mark3)/3
		    RETURN Total
		END FUNCTION

<id> (<parameter>, …)	Calls a procedure or a function.	Add (FirstMark, SecondMark)

Arithmetic operators	
Symbol	**Description**
+	Add
-	Subtract
/	Divide
*	Multiply
^	Exponent
MOD	Modulo
\	Integer division

Relational operators	
Symbol	**Description**
=	equal to
<>	not equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

Logical operators	
Symbol	**Description**
AND	Returns true if both conditions are true.
OR	Returns true if any of the conditions are true.
NOT	Reverses the outcome of the expression; true becomes false, false becomes true.

Assembly language instruction set

Students are not expected to be able to write programs in assembly language. However, they should to be able to explain what a given block of assembly code does.

The table below lists the instructions students need to be familiar with. This is part of the instruction set for the ARM1176JZ-S processor TM. There is no need for them to memorise this table since an appropriate extract from it will be reproduced in the examination paper.

The ARM1176JZ-S processor TM has 16 general-purpose registers (R0–R15), where R13–R15 have special significance. It also has a Current Program Status Register (CPSR) that holds configuration and status bits, including the ALU flags used in conditional execution. You can get more information about this processor at infocenter.arm.com /help/index.jsp.

In the table below general-purpose registers are denoted by the letters 'd', 'n' and 's'.

The #symbol indicates an immediate value.

Instructions		
Operation	**Assembler**	**Action**
Add	ADD {condition} Rd, Rn, #<value>	Adds <value> to the contents of Register n and stores the result in Register d.
	ADD {condition} Rd, Rn, Rm	Adds the contents of Register m to the contents of Register n and stores the result in Register d.
Subtract	SUB {condition} Rd, Rn, #<value>	Subtracts <value> from the contents of Register n and stores the result in Register d.
	SUB {condition} Rd, Rn, Rm	Subtracts the contents of Register m from the contents of Register n and stores the result in Register d.
Multiply	MUL {condition} Rd, Rm, Rs	Multiplies the contents of Register m by the contents of Register s and stores the result in Register d.
Move	MOV {condition} Rd, #<value>	Moves <value> into Register d.
	MOV {condition} Rd, Rm	Moves the contents of Register m into Register d.

Instructions		
Operation	**Assembler**	**Action**
Compare	CMP {condition} Rn, #<value>	Compares <value> with the value in Register n.
	CMP {condition} Rn, Rm	Compares the value in Register m with the value in Register n.
Load	LDR {condition} Rd, [Rm]	Loads the contents of the memory address stored in Register m into Register d.
Branch	B {condition} label	Branch.
	BL {condition} label	Branch and store return address in Register d.
Store	STR {condition} Rd, [Rm]	Stores contents of Register d at memory address stored in Register m.

Conditions	
Mnemonic	**Description**
EQ	Equal
NE	Not equal
GE	Greater than or equal
GT	Greater than
LE	Less than or equal
LT	Less than

Result status flags

There are four result status flags located in the program status register. They are:

N = Negative result

Z = Zero result

C = Carry

V = Overflowed result

Each flag has two possible states. If a flag bit has the value 1, it is said to be true, or set. If it has the value 0, the flag is false or cleared.

The CMP instruction always updates the status of the flags.

An instruction may be executed conditionally based upon the flags set by another instruction, either immediately after the instruction that updated the flags or after any number of intervening instructions that have not updated the flags. In either case, the instruction is only executed if a given combination of flags exists. Otherwise the instruction is ignored.

To make an instruction conditional, a condition code suffix is added to the instruction mnemonic (see conditions table).

An instruction may be unconditional, in which case it executes regardless of the status of the flags.

SQL commands

Command	Description
SELECT (ColumnName, …) FROM TableName [WHERE <condition>] [ORDER BY (ColumnName, …)];	Retrieves data from a table. (The asterisk * can be used if all columns are required in the output.)
SELECT (ColumnName, …) FROM TableName, TableName, … [WHERE <join criteria>] [ORDER BY (ColumnName, …)];	Retrieves data from two related table.
UPDATE TableName SET (ColumnName =Value, …) WHERE <condition>;	Updates rows in a database table.
DELETE FROM TableName [WHERE <condition>];	Deletes rows from a database table.
CREATE TABLE TableName (ColumnName data type, ….);	Creates a database table.
INSERT INTO TableName VALUES (Value, ….);	Inserts rows into a database table.
SELECT ColumnNam) FROM TableName WHERE ColumnName LIKE pattern;	Used to search for a specified pattern in a column.

Items in [ ] brackets are optional.

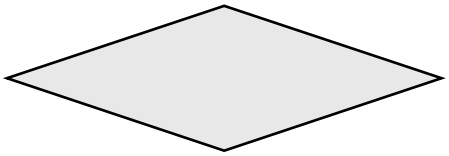
Ellipsis (ie '…') mean the preceding item may be repeated.

Comparison operators	
Symbol	**Description**
=	Equal
<>	Not equal
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal

Flowchart symbols

Denotes the beginning and end of an algorithm.

Denotes a process to be carried out.



Denotes a decision to be made.

Shows the logical flow of the program.

Appendix 3 Codes

Type of code	Use of code	Code number
National classification codes	Every qualification is assigned to a national classification code indicating the subject area to which it belongs. Centres should be aware that students who enter for more than one GCSE qualification with the same classification code will have only one grade (the highest) counted for the purpose of the school and college performance tables.	2610
National Qualifications Framework (NQF) codes	Each qualification title is allocated a Ofqual National Qualifications Framework (NQF) code. The National Qualifications Framework (NQF) code is known as a Qualification Number (QN). This is the code that features in the DfE Section 96 and on the LARA as being eligible for 16-18 and 19+ funding, and is to be used for all qualification funding purposes. The QN is the number that will appear on the student's final certification documentation.	The QN for the qualification in this publication is: 601/0544/6
Paper codes	Each paper is assigned a unit code. This paper code is used as an entry code to indicate that a student wishes to take the assessment for that unit. Centres will need to use the entry codes only when entering students for their examination.	Paper 1: - 1CP0/01 Controlled Assessment: - 1CP0/2A - 1CP0/2B - 1CP0/2C
Cash-in codes	The cash-in code is used as an entry code to aggregate the student's unit scores to obtain the overall grade for the qualification. Centres will need to use the entry codes only when claiming students' qualifications.	GCSE – 1CP0

Appendix 4 Authentication sheet

Controlled Assessment Authentication Statement

This sheet must be completed by the candidate and be provided for work submitted for assessment.

Candidate name:		Centre number:
Candidate number:		Assessor name:
Date issued:	Completion date:	Submitted on:
Qualification:		

Please list the evidence submitted for each task. Indicate the file and folder names where the evidence can be found or describe the nature of the evidence (e.g. video, illustration).

Task ref.	Evidence submitted	File name and location

Comments for note by the Assessor:

Candidate declaration

I certify that the work submitted for this assignment is my own. I have clearly referenced any sources used in the work. I understand that false declaration is a form of malpractice.

Candidate signature: _____ **Date**: _____

Teacher signature: _____ **Date:** _____

Appendix 5 Mapping between CAS and Pearson Edexcel GCSE in Computer Science

CAS curriculum	Location in Pearson Edexcel GCSE
2 — Key concepts	
2.1 Languages, machines and computation	Topic 1: sections 1.1, 1.2 Topic 2: sections 2.1, 2.2, 2.3, 2.4, 2.5, 2.6 Topic 4: sections 4.1, 4.2, 4.3, 4.5
2.2 Data and representation	Topic 2: section 2.3 Topic 3: sections 3.1, 3.2, 3.3, 3.4, 3.5
2.3 Communication and coordination	Topic 5: sections 5.1, 5.2
2.4 Abstraction and design	Topic 1: sections 1.1, 1.2 Topic 2: sections 2.1, 2.2, 2.3, 2.4, 2.5, 2.6 Topic 4: sections 4.1, 4.2, 4.3, 4.4
2.5 Computers and computing are part of a wider context	Topic 6: sections 6.1
3 — Key processes: computational thinking	
3.1 Abstraction: modelling, decomposing and generalising	Topic 1: sections 1.1, 1.2 Topic 2: section 2.6
3.2 Programming	Topic 2: sections 2.1, 2.2, 2.3, 2.4, 2.5, 2.6 Topic 3: section 3.5 Topic 4: section 4.5 Topic 5: section 5.2
4 — Range and content: what a student should know	
4.1 Algorithms	Topic 1: section 1.1 Topic 2: section 2.1
4.2 Programs	Topic 1: sections 1.1, 1.2 Topic 2: sections 2.1, 2.2, 2.3, 2.4, 2.5, 2.6
4.3 Data	Topic 2: section 2.3 Topic 3: sections 3.1, 3.2, 3.3, 3.4, 3.5
4.4 Computers	Topic 4: sections 4.1, 4.2, 4.3, 4.4
4.5 Communication and the internet	Topic 5: sections 5.1, 5.2

Sb180914 G:\WORDPROC\LT\PD\GCSE_LINEAR\9781446914229_GCSE_LIN_COMPSCI_ISSUE2\9781446914229_GCSE_LIN_COMPSCI_ISSUE2.DOC.1–62/2

edexcel

If you have questions regarding this specification, please contact our ICT team:
www.edexcel.com/gcsecomputerscience
TeachingICT@pearson.com
Tel: 0844 372 2186

For general enquiries, contact our customers services team on 0844 576 0027, or via our online support service, Ask the Expert, at **www.edexcel.com/askEdexcel**

For further copies of this publication, please email **customer.orders@pearson.com** or download copies at: **www.edexcel.com/gcsecomputerscience**

For more information on Edexcel and BTEC qualifications please visit our websites: www.edexcel.com and www.btec.co.uk

Pearson Education Limited. Registered in England and Wales No. 872828
Registered Office: Pearson Education Limited, Edinburgh Gate, Harlow, Essex, CM20 2JE
VAT Reg No GB 278 537121

ISBN 978-1-4469-1422-9