

GCSE (9-1)

Computer Science

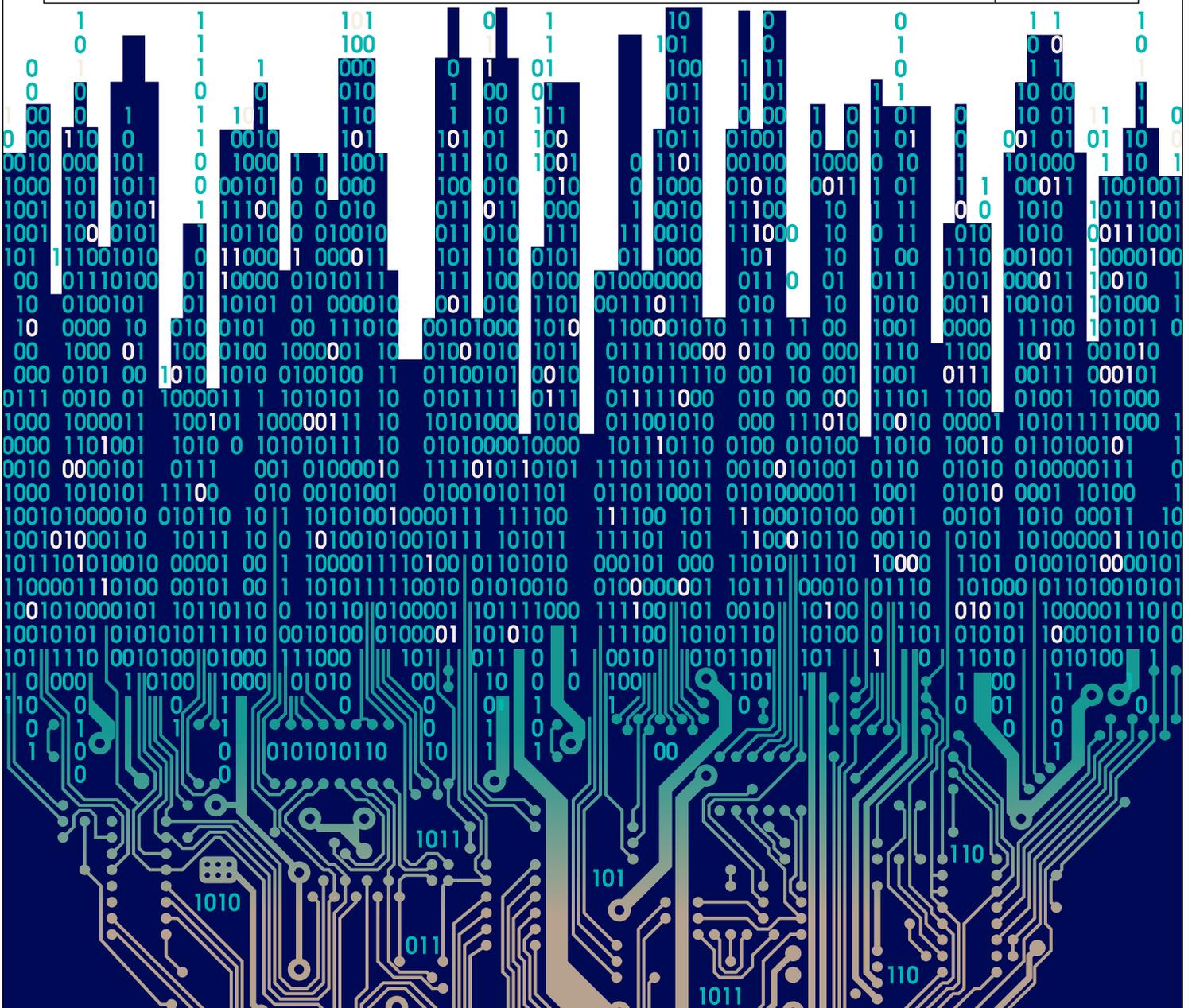
Specification

Pearson Edexcel Level 1/Level 2 GCSE (9-1) in Computer Science (1CP2)

First teaching September 2020

First certification 2022

Issue 1



Contents

1 Introduction	2
Why choose the Pearson Edexcel GCSE Computer Science 2020?	2
Supporting you in planning and implementing this qualification	3
Qualification at a glance	4
2 Subject content	6
Paper 1: Principles of Computer Science	7
Paper 2: Application of Computational Thinking	13
3 Assessment information	18
4 Administration and general information	19
Entries	19
Access arrangements, reasonable adjustments, special consideration and malpractice	20
Student recruitment and progression	23
Appendix 1: Command word taxonomy	27
Appendix 2: Flowchart symbols	29
Appendix 3: The context for the development of this qualification	30
Appendix 4: Codes	32

1 Introduction

Why choose the Pearson Edexcel GCSE Computer Science 2020?

An exciting, practical focus on real-life programming, developing skills relevant to the future.

We have developed our GCSE Computer Science 2020 in collaboration with teachers and the computer science community to create an engaging qualification that equips students with the knowledge and practical skills to thrive in the fast-changing world of computer science.

Our qualification provides a practical approach to developing computational skills. This includes innovative, practical onscreen assessment to ensure all students develop the computational skills they need for an exciting digital future beyond the classroom.

Clear and simple structure

Our qualification has a straightforward structure with six comprehensive topic areas, assessed through two externally-examined papers. One of these is a written paper focused on computational thinking, data, computers, networks, and issues and impact of computing in the world today. The other is a practical onscreen assessment, which focuses on the ability to analyse and solve problems by designing, writing, testing and refining programs.

The qualification's combination of written and practical elements balances theory and practical application, providing students with a rounded experience of computer science.

Practical programming experience

Our approach of assessing programming practically via our new onscreen assessment allows schools to choose which Integrated Development Environment (IDE) to use. This means that students will be able to build their practical skills and experience by programming in a familiar environment.

Clear and accessible assessment for all

The ramping in our papers means they have a gradual increase in difficulty, helping build confidence for students as they work through the questions. The papers' consistent assessment structure and straightforward mark schemes make expectations clear to both teachers and students.

A full support package and expert advice

Our comprehensive teaching and learning support is rich in purposeful activities designed to apply learning to the real world. We provide extensive free and paid for teaching and learning support, designed to support all teachers delivering this qualification.

To help you prepare for the practical onscreen assessment, we will provide a digital repository of programming activities. These will consist of exercises, downloadable data files, sample solutions and commentaries. You can use these programming activities to develop the necessary knowledge and skills.

Our Computer Science Subject Advisor is Tim Brady, a former teacher who is available to offer you the support and advice you need – every step of the way.

Supporting you in planning and implementing this qualification

Planning

- Our **Getting Started** guide gives you an overview of our 2020 GCSE qualification to help you to get to grips with the changes to content and assessment, and to help you understand what these changes mean for you and your students.
- We will give you an editable course planner and scheme of work, including an **interactive scheme of work** that you can adapt to suit your department.

Teaching and learning

There will be a range of free teaching and learning support to help you deliver this qualification, including:

- suggested resource lists
- lesson activities and solutions
- a student guide
- materials for your options evenings.

Preparing for exams

We will also provide a range of resources to help you prepare your students for the assessments, including marked exemplars of student work with examiner commentaries.

ResultsPlus

ResultsPlus provides the most detailed analysis available of your students' exam performance. It can help you identify the topics and skills where further learning would benefit them.

Access to Scripts

Our Access to Scripts service allows you to view your students' marked scripts online or download them in PDF format. Our Access to Script service:

- provides visibility and transparency of marking
- helps to inform post-results decisions for students and boost their confidence
- helps identify skills gaps, so you can tailor future teaching plans and refine approaches
- supports departmental CPD.

Get help and support

Our Subject Advisor service and online community, led by Tim Brady, will ensure you receive help and guidance from us, and that you can share ideas and information with other teachers. You can sign up to receive e-newsletters from Tim to keep up to date with qualification updates and product and service news.

Subject Advisor contact details

UK: 0333 016 4160

International: +44 (0) 333 016 4160

Contact us at: <https://support.pearson.com/uk/s/qualification-contactus>

Learn more at [qualifications.pearson.com](https://www.pearson.com/qualifications)

Qualification at a glance

Content and assessment overview

The Pearson Edexcel Level 1/Level 2 GCSE (9–1) in Computer Science consists of two externally-examined papers.

Paper 1 is a written examination and Paper 2 is a practical onscreen assessment.

Paper 1: Principles of Computer Science (*Paper code: 1CP2/01)
<i>Written examination: 1 hour and 30 minutes</i> <i>50% of the qualification</i> <i>75 marks</i>
Content overview This paper will assess Topics 1 to 5. <ul style="list-style-type: none">• Topic 1: Computational thinking – understanding of what algorithms are, what they are used for and how they work; ability to follow, amend and write algorithms; ability to construct truth tables.• Topic 2: Data – understanding of binary, data representation, data storage and compression.• Topic 3: Computers – understanding of hardware and software components of computer systems and characteristics of programming languages.• Topic 4: Networks – understanding of computer networks and network security.• Topic 5: Issues and impact – awareness of emerging trends in computing technologies, and the impact of computing on individuals, society and the environment, including ethical, legal and ownership issues.
Assessment overview This paper consists of five compulsory questions, each one focused on one of the topic areas. The questions consist of multiple-choice, short-, medium- and extended-open-response, tabular and diagrammatic items.

Paper 2: Application of Computational Thinking (*Paper code: 1CP2/02)

Onscreen examination: 2 hours

50% of the qualification

75 marks

Content overview

This paper will assess Topic 6: Problem solving with programming.

The main focus of this paper is:

- understanding what algorithms are, what they are used for and how they work in relation to creating programs
- understanding how to decompose and analyse problems
- ability to read, write, refine and evaluate programs.

Assessment overview

This practical paper requires students to design, write, test and refine programs in order to solve problems.

Students will complete this assessment onscreen using their Integrated Development Environment (IDE) of choice.

They will be provided with:

- coding files
- a hard copy of the question paper
- the Programming Language Subset (PLS) – as an insert in the question paper and in electronic format.

Students should then answer the questions onscreen using Python 3.

This assessment consists of six compulsory questions.

*See *Appendix 4: Codes* for a description of this code and all other codes relevant to this qualification.

2 Subject content

Qualification aims and objectives

The aims and objectives of this qualification are to enable students 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
- 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 impact of digital technology on wider society, including issues of privacy and cybersecurity
- apply mathematical skills relevant to computer science.

Paper 1: Principles of Computer Science

Overview

In order to become proficient computer scientists, it is essential that students have knowledge and understanding of the field's fundamental principles and concepts.

Content

Topic 1: Computational thinking

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

Students are expected to be familiar with and use the Programming Language Subset (PLS) document provided on the [GCSE in Computer Science section of our website](#).

The flowchart symbols students are expected to be familiar with and use are shown in *Appendix 2*.

Subject content	Students should:
1.1 Decomposition and abstraction	1.1.1 understand the benefit of using decomposition and abstraction to model aspects of the real world and analyse, understand and solve problems
	1.1.2 understand the benefits of using subprograms
1.2 Algorithms	1.2.1 be able to follow and write algorithms (flowcharts, pseudocode*, program code) that use sequence, selection, repetition (count-controlled, condition-controlled) and iteration (over every item in a data structure), and input, processing and output to solve problems
	1.2.2 understand the need for and be able to follow and write algorithms that use variables and constants and one- and two-dimensional data structures (strings, records, arrays)
	1.2.3 understand the need for and be able to follow and write algorithms that use arithmetic operators (addition, subtraction, division, multiplication, modulus, integer division, exponentiation), relational operators (equal to, less than, greater than, not equal to, less than or equal to, greater than or equal to) and logical operators (AND, OR, NOT)
	1.2.4 be able to determine the correct output of an algorithm for a given set of data and use a trace table to determine what value a variable will hold at a given point in an algorithm
	1.2.5 understand types of errors that can occur in programs (syntax, logic, runtime) and be able to identify and correct logic errors in algorithms
	1.2.6 understand how standard algorithms (bubble sort, merge sort, linear search, binary search) work

Subject content	Students should:
	1.2.7 be able to use logical reasoning and test data to evaluate an algorithm's fitness for purpose and efficiency (number of compares, number of passes through a loop, use of memory)
1.3 Truth tables	1.3.1 be able to apply logical operators (AND, OR, NOT) in truth tables with up to three inputs to solve problems

*In this specification, the term 'pseudocode' is used to denote an informal written description of a program. Pseudocode uses imprecise English language statements and does not require any strict programming syntax.

Topic 2: Data

Computers use binary to represent different types of data.

Students are expected to learn how different types of data are represented in a computer.

Subject content	Students should:
2.1 Binary	2.1.1 understand that computers use binary to represent data (numbers, text, sound, graphics) and program instructions and be able to determine the maximum number of states that can be represented by a binary pattern of a given length
	2.1.2 understand how computers represent and manipulate unsigned integers and two's complement signed integers
	2.1.3 be able to convert between denary and 8-bit binary numbers (0 to 255, -128 to +127)
	2.1.4 be able to add together two positive binary patterns and apply logical and arithmetic binary shifts
	2.1.5 understand the concept of overflow in relation to the number of bits available to store a value
	2.1.6 understand why hexadecimal notation is used and be able to convert between hexadecimal and binary
2.2 Data representation	2.2.1 understand how computers encode characters using 7-bit ASCII
	2.2.2 understand how bitmap images are represented in binary (pixels, resolution, colour depth)
	2.2.3 understand how analogue sound is represented in binary (amplitude, sample rate, bit depth, sample interval)
	2.2.4 understand the limitations of binary representation of data when constrained by the number of available bits
2.3 Data storage and compression	2.3.1 understand that data storage is measured in binary multiples (bit, nibble, byte, kibibyte, mebibyte, gibibyte, tebibyte) and be able to construct expressions to calculate file sizes and data capacity requirements
	2.3.2 understand the need for data compression and methods of compressing data (lossless, lossy)

Topic 3: Computers

Students must be familiar with the hardware and software components that make up a computer system.

Subject content	Students should:
3.1 Hardware	3.1.1 understand the von Neumann stored program concept and the role of main memory (RAM), CPU (control unit, arithmetic logic unit, registers), clock, address bus, data bus, control bus in the fetch-decode-execute cycle
	3.1.2 understand the role of secondary storage and the ways in which data is stored on devices (magnetic, optical, solid state)
	3.1.3 understand the concept of an embedded system and what embedded systems are used for
3.2 Software	3.2.1 understand the purpose and functionality of an operating system (file management, process management, peripheral management, user management)
	3.2.2 understand the purpose and functionality of utility software (file repair, backup, data compression, disk defragmentation, anti-malware)
	3.2.3 understand the importance of developing robust software and methods of identifying vulnerabilities (audit trails, code reviews)
3.3 Programming languages	3.3.1 understand the characteristics and purposes of low-level and high-level programming languages
	3.3.2 understand how an interpreter differs from a compiler in the way it translates high-level code into machine code

Topic 4: Networks

Most computer applications in use today would not be possible without networks. Students should understand the key principles behind the organisation of computer networks.

Subject content	Students should:
4.1 Networks	4.1.1 understand why computers are connected in a network
	4.1.2 understand different types of networks (LAN, WAN)
	4.1.3 understand how the internet is structured (IP addressing, routers)
	4.1.4 understand how the characteristics of wired and wireless connectivity impact on performance (speed, range, latency, bandwidth)
	4.1.5 understand that network speeds are measured in bits per second (kilobit, megabit, gigabit) and be able to construct expressions involving file size, transmission rate and time
	4.1.6 understand the role of and need for network protocols (Ethernet, Wi-Fi, TCP/IP, HTTP, HTTPS, FTP) and email protocols (POP3, SMTP, IMAP)
	4.1.7 understand how the 4-layer (application, transport, internet, link) TCP/IP model handles data transmission over a network
	4.1.8 understand characteristics of network topologies (bus, star, mesh)
4.2 Network security	4.2.1 understand the importance of network security, ways of identifying network vulnerabilities (penetration testing, ethical hacking) and methods of protecting networks (access control, physical security, firewalls)

Topic 5: Issues and impact

Students should be aware of the influence of digital technology and recognise some of the issues and the impact on wider society associated with its use.

Subject content	Students should:
5.1 Environmental	5.1.1 understand environmental issues associated with the use of digital devices (energy consumption, manufacture, replacement cycle, disposal)
5.2 Ethical and legal	5.2.1 understand ethical and legal issues associated with the collection and use of personal data (privacy, ownership, consent, misuse, data protection)
	5.2.2 understand ethical and legal issues associated with the use of artificial intelligence, machine learning and robotics (accountability, safety, algorithmic bias, legal liability)
	5.2.3 understand methods of intellectual property protection for computer systems and software (copyright, patents, trademarks, licencing)
5.3 Cybersecurity	5.3.1 understand the threat to digital systems posed by malware (viruses, worms, Trojans, ransomware, key loggers) and how hackers exploit technical vulnerabilities (unpatched software, out-of-date anti-malware) and use social engineering to carry out cyberattacks
	5.3.2 understand methods of protecting digital systems and data (anti-malware, encryption, acceptable use policies, backup and recovery procedures)

Assessment information

- First assessment: May/June 2022.
- The paper is 1 hour and 30 minutes.
- The paper consists of five questions (one question per topic).
- The paper is out of 75 marks.
- Students must answer all questions.
- The paper consists of multiple-choice, short-, medium- and extended-open-response, tabular and diagrammatic items.
- The paper will include questions that target mathematics at Level 2.
- The paper will include questions that target computer-related mathematics.
- Calculators must not be used in the examination.

Sample assessment materials

A sample question paper and mark scheme for this paper can be found in the *Pearson Edexcel Level 1/Level 2 GCSE (9–1) in Computer Science Sample Assessment Materials (SAMs)* document.

The SAMs will be available on the [GCSE in Computer Science section of our website](#). Please note that the order of the topics will vary in each live assessment. A full list of command words that will be used in assessments are shown in *Appendix 1: Command word taxonomy*.

Paper 2: Application of Computational Thinking

Overview

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

Content

Topic 6: Problem solving with programming

All problems set in the practical programming tasks in the examination can be solved with the functionalities presented in the Programming Language Subset (PLS) document provided on the [GCSE in Computer Science section of our website](#).

Subject content	Students should:
6.1 Develop code	6.1.1 be able to use decomposition and abstraction to analyse, understand and solve problems
	6.1.2 be able to read, write, analyse and refine programs written in a high-level programming language
	6.1.3 be able to convert algorithms (flowcharts, pseudocode*) into programs
	6.1.4 be able to use techniques (layout, indentation, comments, meaningful identifiers, white space) to make programs easier to read, understand and maintain
	6.1.5 be able to identify, locate and correct program errors (logic, syntax, runtime)
	6.1.6 be able to use logical reasoning and test data to evaluate a program's fitness for purpose and efficiency (number of compares, number of passes through a loop, use of memory)

*In this specification, the term 'pseudocode' is used to denote an informal written description of a program. Pseudocode uses imprecise English language statements and does not require any strict programming syntax.

Subject content	Students should:
6.2 Constructs	6.2.1 understand the function of and be able to identify the structural components of programs (constants, variables, initialisation and assignment statements, command sequences, selection, repetition, iteration, data structures, subprograms, parameters, input/output)
	6.2.2 be able to write programs that make appropriate use of sequencing, selection, repetition (count-controlled, condition-controlled), iteration (over every item in a data structure) and single entry/exit points from code blocks and subprograms
6.3 Data types and structures	6.3.1 be able to write programs that make appropriate use of primitive data types (integer, real, Boolean, char) and one- and two-dimensional structured data types (string, array, record)
	6.3.2 be able to write programs that make appropriate use of variables and constants
	6.3.3 be able to write programs that manipulate strings (length, position, substrings, case conversion)
6.4 Input/output	6.4.1 be able to write programs that accept and respond appropriately to user input
	6.4.2 be able to write programs that read from and write to comma separated value text files
	6.4.3 understand the need for and be able to write programs that implement validation (length check, presence check, range check, pattern check)
	6.4.4 understand the need for and be able to write programs that implement authentication (ID and password, lookup)
6.5 Operators	6.5.1 be able to write programs that use arithmetic operators (addition, subtraction, division, multiplication, modulus, integer division, exponentiation)
	6.5.2 be able to write programs that use relational operators (equal to, less than, greater than, not equal to, less than or equal to, greater than or equal to)
	6.5.3 be able to write programs that use logical operators (AND, OR, NOT)
6.6 Subprograms	6.6.1 be able to write programs that use pre-existing (built-in, library) and user-devised subprograms (procedures, functions)
	6.6.2 be able to write functions that may or may not take parameters but must return values, and procedures that may or may not take parameters but do not return values
	6.6.3 understand the difference between and be able to write programs that make appropriate use of global and local variables

Onscreen problem solving with programming

Students should engage with a broad range of activities to enable them to learn to program to the standard required for this qualification.

Mastering programming involves much more than simply learning the syntax/semantics of a programming language. It also involves learning strategies for problem solving, embracing mistakes as opportunities to learn, mastering a few simple tools and working together with others to achieve goals.

We have adopted Python programming language as the vehicle to explore, learn and assess the aspects of problem solving and programming covered in this qualification. Python is popular and commonly used in education.

The requirements of this qualification can be met using the Python 3 programming language. Centres are advised to choose the most up-to-date version of an interpreter that supports Python 3. There are many freely available Python 3 interpreters, often installable as part of an integrated development environment (IDE).

Programming environments

Online guided tutorials are useful in motivating and engaging students. They normally present small challenges where students copy or reproduce code to achieve a short-term goal, often based on moving up levels. Using these environments exclusively may not prepare students for the type of problems they will encounter in Paper 2. Teachers should consider highlighting how each level relates directly to the requirements of the specification. For example, if students are directing a tank to move forward until it encounters an obstacle using a while loop, then the corresponding structure of the while loop could be explored in a non-graphical text-based environment, including on paper.

Students will need experience using an IDE to write, run and debug their programming code. Developing skills in using an IDE will allow students to be more productive, especially in time-constrained activities. A suitable IDE includes these functionalities:

- an editor, with line numbers
- a syntax highlighter and checker
- breakpoints and stepping
- a variable inspector.

There are many IDEs available to freely download online. Teachers are advised to evaluate several before selecting one suitable for their students.

Classroom strategies

Several professional organisations exist for computing and computer science teachers. It is advisable to join one or more of these organisations to keep up to date and share current best practice in the teaching of computer science. A range of strategies that teachers could consider in teaching programming include:

- unplugged activities, which focus on reading and tracing code before attempting to write code
- correcting errors in code, both on paper and in an IDE
- an approach such as PRIMM (predict, run, investigate, modify, make) to build on small steps
- paired programming to write and debug code
- faded worked examples, where scaffolding is slowly removed

- peer instruction to encourage students to learn from each other and develop skills in reading code
- Parsons problems, where code must be arranged in order, reducing the need for copying or typing in code
- showing consistent exemplar code, using the good programming practices set out in the Programming Language Subset (PLS).

Information on any unfamiliar strategies can be found online, using a search engine.

Programming Language Subset (PLS)

Both teachers and students are encouraged to look at the Programming Language Subset (PLS). The PLS represents a specific set of constructs that are sufficient to successfully answer any question in Paper 2. Any student successfully using more esoteric or complex constructs or approaches not included in the PLS will still be awarded marks in Paper 2 if the solution is valid. The constructs contained in the PLS are found in most high-level programming languages and therefore form a foundation of computer science knowledge required in the assessment.

The PLS will be valid for the lifetime of the qualification. However, if updates are required, we will ensure the latest version is published by the 31st of January in the year of the examination. Please note that students will be given both printed and electronic versions of the PLS during the examination. Please see the *Pearson Edexcel Level 1/Level 2 GCSE (9-1) in Computer Science – Paper 2 Instructions for the Conduct of the Examination (ICE)* for further information. The PLS and ICE are available to download on the [GCSE in Computer Science section of our website](#).

Teacher support

We will develop a range of teaching and learning materials to help centres prepare for the onscreen assessment of problem solving with programming. These include a good programming practice guide elaborating on and exemplifying the constructs in the PLS, and programming exercises with commentaries and solutions. The exercises can be used by teachers either as classroom activities or for formative assessment purposes. These teacher support materials will be made available on the [GCSE in Computer Science section of our website](#).

Practical Programming Statement (PPS)

Centres are required to complete a Practical Programming Statement (PPS), which confirms they have taken reasonable steps to ensure that each student sitting our GCSE in Computer Science has had the opportunity to demonstrate their programming skills in terms of design, write, test and refine programs in Python during their course of study. The PPS should be completed by a member of the senior leadership team at the centre. Failure by a centre to provide a completed PPS to us in a timely manner will result in potential malpractice and/or maladministration.

The PPS will be made available by the 31st of January in the year of the examination on the [GCSE in Computer Science section of our website](#).

The completed PPS must be submitted to us by the 31st of May in the year of the examination. Centres are reminded that if they do not submit this form it may result in malpractice/maladministration.

Instructions for the conduct of the examination (ICE)

For detailed information on the set up of candidate secure user areas, download of coding files, administration of the examination and transmission of candidates' completed coding

files to us for marking, please refer to the *Pearson Edexcel Level 1/Level 2 GCSE (9-1) in Computer Science – Paper 2 Instructions for the Conduct of the Examination (ICE)*.

This document will be updated on an annual basis so that in each year of examination the version to be used will be available by the 31st of January. It is available on the [GCSE in Computer Science section of our website](#).

Security and backups

For materials stored electronically, 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 ensured following the examination.

Assessment information

- First assessment: May/June 2022.
- The paper is 2 hours.
- The paper consists of six compulsory questions.
- The paper is out of 75 marks.
- Students must answer all questions.
- The questions are practical in nature and require students to design, write, test and refine programs in order to solve problems.
- Students will complete this assessment onscreen using their Integrated Development Environment (IDE) of choice.
- Students will be provided with coding files, a hard copy of the question paper, and the Programming Language Subset (PLS) document. Students should then answer the questions onscreen using Python 3.
- Students must not have access to the internet.
- The paper will include questions that target computer-related mathematics.
- The paper will include questions that target mathematics at Level 2.

Synoptic assessment

Synoptic assessment requires students to work across different parts of a qualification by applying their accumulated knowledge and understanding of multiple topic or subject areas to practical contexts.

Synoptic assessment enables students to show their ability to combine their skills, knowledge and understanding within the breadth and depth of the subject.

Synoptic assessment will be tested in Paper 2.

Sample assessment materials

A sample question paper and mark scheme can be found in the *Pearson Edexcel Level 1/Level 2 GCSE (9-1) in Computer Science Sample Assessment Materials (SAMs)* document.

The SAMs will be available on the [GCSE in Computer Science section of our website](#). Coding files for the Paper 2 SAMs will also be available for download. These include the original coding files that accompany the sample question paper, and exemplar finished coding files to accompany the mark scheme. A copy of the PLS will also be available.

A full list of command words that will be used in assessments are shown in *Appendix 1: Command word taxonomy*.

3 Assessment information

Assessment Objectives

Students must:		% in GCSE
AO1	Demonstrate knowledge and understanding of the key concepts and principles of computer science	30
AO2	Apply knowledge and understanding of key concepts and principles of computer science	40
AO3	Analyse problems in computational terms: <ul style="list-style-type: none"> • to make reasoned judgements • to design, program, evaluate and refine solutions. 	30
Total		100

Breakdown of Assessment Objectives

Paper	Assessment Objectives			Total % for all Assessment Objectives
	AO1 %	AO2 %	AO3 %	
Paper 1: Principles of Computer Science	30	20	0	50
Paper 2: Application of Computational Thinking	0	20	30	50
Total for GCSE	30	40	30	100

4 Administration and general information

Entries

Details of how to enter students for the examinations for this qualification can be found in our *UK Information Manual*. A copy is made available to all examinations officers and is available on our website: qualifications.pearson.com

Re-taking the qualification

Students wishing to re-take this qualification are required to re-take both external papers in May/June in any single year.

Discount code and performance tables

Centres should be aware that students who enter for more than one GCSE, or other Level 2 qualifications with the same discount code, will have only the grade for their 'first entry' counted for the purpose of the school and college performance tables (please see *Appendix 4: Codes*). For further information about what constitutes 'first entry' and full details of how this policy is applied, please refer to the DfE website: www.gov.uk/government/organisations/department-for-education

Students should be advised that if they take two GCSEs with the same discount code, schools and colleges they wish to progress to are likely to take the view that this achievement is equivalent to only one GCSE. The same view may be taken if students take two GCSE or other Level 2 qualifications that have different discount codes but which have significant overlap of content. Students or their advisers who have any doubts about their subject combinations should check with the institution they wish to progress to before embarking on their programmes.

Access arrangements, reasonable adjustments, special consideration and malpractice

Equality and fairness are central to our work. Our equality policy requires all students to have equal opportunity to access our qualifications and assessments, and our qualifications to be awarded in a way that is fair to every student.

We are committed to making sure that:

- students with a protected characteristic (as defined by the Equality Act 2010) are not, when they are undertaking one of our qualifications, disadvantaged in comparison to students who do not share that characteristic
- all students achieve the recognition they deserve for undertaking a qualification and that this achievement can be compared fairly to the achievement of their peers.

Language of assessment

Assessment of this qualification will be available in English. All student work must be in English.

Access arrangements

Access arrangements are agreed before an assessment. They allow students with special educational needs, disabilities or temporary injuries to:

- access the assessment
- show what they know and can do without changing the demands of the assessment.

The intention behind an access arrangement is to meet the particular needs of an individual student with a disability, without affecting the integrity of the assessment. Access arrangements are the principal way in which awarding bodies comply with the duty under the Equality Act 2010 to make 'reasonable adjustments'.

Access arrangements should always be processed at the start of the course. Students will then know what is available and have the access arrangement(s) in place for assessment.

Reasonable adjustments

The Equality Act 2010 requires an awarding organisation to make reasonable adjustments where a person with a disability would be at a substantial disadvantage in undertaking an assessment. The awarding organisation is required to take reasonable steps to overcome that disadvantage.

A reasonable adjustment for a particular person may be unique to that individual and therefore might not be in the list of available access arrangements.

Whether an adjustment will be considered reasonable will depend on a number of factors, which will include:

- the needs of the student with the disability
- the effectiveness of the adjustment
- the cost of the adjustment; and
- the likely impact of the adjustment on the student with the disability and other students.

An adjustment will not be approved if it involves unreasonable costs to the awarding organisation, timeframes or affects the security or integrity of the assessment. This is because the adjustment is not 'reasonable'.

Special consideration

Special consideration is a post-examination adjustment to a student's mark or grade to reflect temporary injury, illness or other indisposition at the time of the assessment, which has had, or is reasonably likely to have had, a material effect on a candidate's ability to take an assessment or demonstrate their level of attainment in an assessment.

Further information

Please see our website for further information about how to apply for access arrangements and special consideration.

For further information about access arrangements, reasonable adjustments and special consideration, please refer to the JCQ website: www.jcq.org.uk.

Malpractice

Candidate malpractice

Candidate malpractice refers to any act by a candidate that compromises or seeks to compromise the process of assessment or which undermines the integrity of the qualifications or the validity of results/certificates.

Candidate malpractice in examinations **must** be reported to us on a *JCQ Form M1* (available at www.jcq.org.uk/exams-office/malpractice). The form should be emailed to candidatemalpractice@pearson.com. Please provide as much information and supporting documentation as possible. Note that the final decision regarding appropriate sanctions lies with us.

Failure to report malpractice constitutes staff or centre malpractice.

Staff/centre malpractice

Staff and centre malpractice includes both deliberate malpractice and maladministration of our qualifications. As with candidate malpractice, staff and centre malpractice is any act that compromises or seeks to compromise the process of assessment or which undermines the integrity of the qualifications or the validity of results/certificates.

All cases of suspected staff malpractice and maladministration **must** be reported immediately, before any investigation is undertaken by the centre, to us on a *JCQ Form M2(a)* (available at www.jcq.org.uk/exams-office/malpractice). The form, supporting documentation and as much information as possible should be emailed to pqsmalpractice@pearson.com. Note that the final decision regarding appropriate sanctions lies with us.

Failure to report malpractice itself constitutes malpractice.

More-detailed guidance on malpractice can be found in the latest version of the document *General and Vocational Qualifications Suspected Malpractice in Examinations and Assessments Policies and Procedures*, available at www.jcq.org.uk/exams-office/malpractice.

Awarding and reporting

This qualification will be graded, awarded and certificated to comply with the requirements of Ofqual's General Conditions of Recognition.

This GCSE qualification will be graded and certificated on a nine-grade scale from 9 to 1, using the total subject mark where 9 is the highest grade. Individual papers are not graded.

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.

The first certification opportunity for this qualification is 2022.

Student recruitment and progression

Pearson follows the JCQ policy concerning recruitment to our qualifications in 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 learning and other requirements

There are no prior learning or other requirements for this qualification.

Progression

Students can progress from this qualification to:

- further studies, for example A Levels, a BTEC in Computer Science
- employment, where further training may be available.

Appendices

Appendix 1: Command word taxonomy	27
Appendix 2: Flowchart symbols	29
Appendix 3: The context for the development of this qualification	30
Appendix 4: Codes	32

Appendix 1: Command word taxonomy

The following tables show the list of command verbs and their definitions for use in the question papers. Only the verbs listed in these tables will be used in question papers. Please note that the command words listed for Paper 1 will not all be used in each examination series.

Paper 1

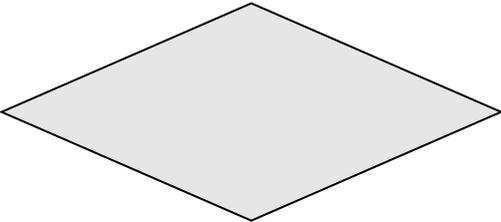
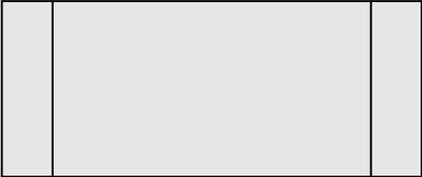
Command word	Mark Tariff	Definition
Amend	1 to 6 marks	Requires changes or additions to code, or deletions or rearrangement of code.
Calculate	2 to 4 marks	Obtain a numerical answer, showing relevant working. If the answer has a unit, this must be included.
Complete	2 to 6 marks	Requires the completion of a table/diagram/algorithm (in any notation).
Construct	2 to 4 marks	Usually requires creation of an artefact using subject-specific symbolic representation, rules and syntax.
Convert	2 to 6 marks	Requires changing information from one symbolic representation to another representation. May require amending to provide new functionality/facility.
Define	1 to 2 marks	When the meaning of a term is expected but there are different ways of how this can be described.
Describe	2 to 4 marks	To give an account of something. Statements in the response need to be developed, as they are often linked, but do not need to include a justification or reason.
Discuss	6 marks	<ul style="list-style-type: none"> Identify the issue/situation/problem/argument that is being assessed within the question. Explore all aspects of an issue/situation/problem/argument. Investigate the issue/situation by reasoning or argument.
Draw	2 to 6 marks	Produce a diagram/image either using a ruler or freehand. May require labelling/annotation to express meaning. Used when symbolic representations need to be manipulated.
Explain	2 to 4 marks	An explanation requires a justification/exemplification of a point. The answer must contain some element of reasoning/justification, this can include mathematical/logical explanations. The mark scheme will have marking points which are linked.
Give/State/Name	1 mark (for each)	All these command words are really synonyms. They all require recall of one or more pieces of information.
Identify	1 mark (for each)	Usually requires some key information to be selected from a given stimulus/resource/set of options.
Write	1 to 6 marks	Usually requires creation/manipulation of an artefact using a subject-specific notation.

Paper 2

Command word	Mark Tariff	Definition
Amend	5 to 15 marks	Requires changes or additions to code, or deletions or rearrangement of code.
Write	5 to 15 marks	Usually requires creation/manipulation of a program using a high-level programming language.

Appendix 2: Flowchart symbols

Students are expected to be familiar with and use the following flowchart symbols for *Paper 1: Principles of Computer Science*. Students can answer any/all questions using these symbols but if they use other symbols that are valid and understandable in their flowchart, they will be given credit and will gain marks. Whenever we phrase a question, we will only use these symbols.

	Denotes the start and end of an algorithm.
	Denotes a process to be carried out.
	Denotes an input or output.
	Denotes a decision to be made.
	Shows the logical flow of the program.
	Denotes a pre-defined subprogram.

Appendix 3: The context for the development of this qualification

All our qualifications are designed to meet our World Class Qualification Principles^[1] and our ambition to put the student at the heart of everything we do.

We have developed and designed this qualification by:

- reviewing other curricula and qualifications to ensure that it is comparable with those taken in high-performing jurisdictions overseas
- consulting with key stakeholders on content and assessment, including learned bodies, subject associations, higher-education academics and teachers to ensure this qualification is suitable for a UK context
- reviewing the legacy qualification and building on its positive attributes.

This qualification has also been developed to meet criteria stipulated by Ofqual in their documents *GCSE (9 to 1) Qualification Level Conditions and Requirements* and *GCSE Subject Level Conditions and Requirements for Computer Science*, published in April 2014 and February 2019 respectively.

^[1] Pearson's World Class Qualification Principles ensure that our qualifications are:

- **demanding**, through internationally benchmarked standards, encouraging deep learning and measuring higher-order skills
- **rigorous**, through setting and maintaining standards over time, developing reliable and valid assessment tasks and processes, and generating confidence in end users of the knowledge, skills and competencies of certified students
- **inclusive**, through conceptualising learning as continuous, recognising that students develop at different rates and have different learning needs, and focusing on progression
- **empowering**, through promoting the development of transferable skills.

From Pearson's Expert Panel for World Class Qualifications

May 2014

"The reform of the qualifications system in England is a profoundly important change to the education system. Teachers need to know that the new qualifications will assist them in helping their learners make progress in their lives.

When these changes were first proposed we were approached by Pearson to join an 'Expert Panel' that would advise them on the development of the new qualifications.

We were chosen, either because of our expertise in the UK education system, or because of our experience in reforming qualifications in other systems around the world as diverse as Singapore, Hong Kong, Australia, and a number of countries across Europe.

We have guided Pearson through what we judge to be a rigorous qualification development process that has included:

- Extensive international comparability of subject content against the highest-performing jurisdictions in the world
- Benchmarking assessments against UK and overseas providers to ensure that they are at the right level of demand
- Establishing External Subject Advisory Groups, drawing on independent subject-specific expertise to challenge and validate our qualifications
- Subjecting the final qualifications to scrutiny against the DfE content and Ofqual accreditation criteria in advance of submission.

Importantly, we have worked to ensure that the content and learning is future oriented. The design has been guided by what is called an 'Efficacy Framework', meaning learner outcomes have been at the heart of this development throughout.

We understand that ultimately it is excellent teaching that is the key factor to a learner's success in education. As a result of our work as a panel we are confident that we have supported the development of qualifications that are outstanding for their coherence, thoroughness and attention to detail and can be regarded as representing world-class best practice."

Sir Michael Barber (Chair)

Chief Education Advisor, Pearson plc

Professor Lee Sing Kong

Director, National Institute of Education, Singapore

Bahram Bekhradnia

President, Higher Education Policy Institute

Professor Jonathan Osborne

Stanford University

Dame Sally Coates

Principal, Burlington Danes Academy

Professor Dr Ursula Renold

Federal Institute of Technology, Switzerland

Professor Robin Coningham

Pro-Vice Chancellor, University of Durham

Professor Bob Schwartz

Harvard Graduate School of Education

Dr Peter Hill

Former Chief Executive ACARA

All titles correct as at May 2014

Appendix 4: Codes

Type of code	Use of code	Code
Discount codes	Every qualification eligible for performance tables is assigned a discount code indicating the subject area to which it belongs. Discount codes are published by the DfE.	Please see the GOV.UK website*
Regulated Qualifications Framework (RQF) codes	Each qualification title is allocated an Ofqual Regulated Qualifications Framework (RQF) code. The RQF 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 will appear on students' final certification documentation.	The QN for this qualification is: 601/8058/4
Subject codes	The subject code is used by centres to enter students for a qualification. Centres will need to use the entry codes only when claiming students' qualifications.	GCSE – 1CP2
Paper codes	These codes are provided for reference purposes. Students do not need to be entered for individual papers.	Paper 1: 1CP2/01 Paper 2: 1CP2/02

*<https://www.gov.uk/government/publications/key-stage-4-qualifications-discount-codes-and-point-scores>

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 website at qualifications.pearson.com. Alternatively, you can get in touch with us using the details on our contact us page at qualifications.pearson.com/contactus

About Pearson

Pearson is the world's leading learning company, with 35,000 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 qualifications.pearson.com

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

All information in this specification is correct at time of publication.

ISBN 978 1 446 96579 5

All the material in this publication is copyright

© Pearson Education Limited 2020

For information about Pearson Qualifications, including Pearson Edexcel, BTEC and LCCI qualifications visit qualifications.pearson.com

Edexcel and BTEC are registered trademarks of Pearson Education Limited

Pearson Education Limited. Registered in England and Wales No. 872828
Registered Office: 80 Strand, London WC2R 0RL

VAT Reg No GB 278 537121

© Valery Brozhinsky/Shutterstock



ISBN 978-1-4469-6579-5
9 781446 965795 >