CAMBRIDGE
International Examinations

Cambridge
O Level

SYLLABUS

Cambridge O Level
Computer Science
2210

For examination in June and November 2017, 2018 and 2019

Cambridge Secondary 2

Version 1

Contents

1.1 Why choose Cambridge?

Cambridge International Examinations is part of the University of Cambridge. We prepare school students for life, helping them develop an informed curiosity and a lasting passion for learning. Our international qualifications are recognised by the world's best universities and employers, giving students a wide range of options in their education and career. As a not-for-profit organization, we devote our resources to delivering high-quality educational programs that can unlock learners' potential.
Our programs set the global standard for international education. They are created by subject experts, are rooted in academic rigour, and provide a strong platform for progression. Over 10 000 schools in 160 countries work with us to prepare nearly a million learners for their future with an international education from Cambridge.

Cambridge learners

Cambridge programs and qualifications develop not only subject knowledge but also skills. We encourage Cambridge learners to be:

- **confident** in working with information and ideas – their own and those of others
- **responsible** for themselves, responsive to and respectful of others
- **reflective** as learners, developing their ability to learn
- **innovative** and equipped for new and future challenges
- **engaged** intellectually and socially, ready to make a difference.

1.2 Why choose Cambridge O Level?

Cambridge O Levels have been designed for an international audience and are sensitive to the needs of different countries. These qualifications are designed for learners whose first language may not be English and this is acknowledged throughout the examination process. The Cambridge O Level syllabus also allows teaching to be placed in a localized context, making it relevant in varying regions.
Our aim is to balance knowledge, understanding and skills in our programs and qualifications to enable students to become effective learners and to provide a solid foundation for their continuing educational journey.
Through our professional development courses and our support materials for Cambridge O Levels, we provide the tools to enable teachers to prepare learners to the best of their ability and work with us in the pursuit of excellence in education.
Cambridge O Levels are considered to be an excellent preparation for Cambridge International AS and A Levels, the Cambridge AICE (Advanced International Certificate of Education) Group Award, Cambridge Pre-U, and other education programs, such as the US Advanced Placement program and the International Baccalaureate Diploma program. Learn more about Cambridge O Levels at www.cie.org.uk/cambridgesecondary2

Guided learning hours

Cambridge O Level syllabuses are designed on the assumption that learners have about 130 guided learning hours per subject over the duration of the course, but this is for guidance only. The number of hours required to gain the qualification may vary according to local curricular practice and the learners' prior experience of the subject.

1.3 Why choose Cambridge O Level Computer Science?

Computer science is the study of the foundational principles and practices of computation and computational thinking and their application in the design and development of computer systems. Learning computational thinking involves learning to program, that is to write computer code, because this is the means by which computational thinking is expressed.

Cambridge O Level Computer Science enables learners to develop an interest in computing and to gain confidence in computational thinking and programming. They develop their understanding of the main principles of problem-solving using computers.

Learners apply their understanding to develop computer-based solutions to problems using algorithms and a high-level programming language. They also develop a range of technical skills, as well as the ability to test effectively and to evaluate computing solutions.

This qualification will help learners appreciate current and emerging computing technologies and the benefits of their use. They learn to recognise the ethical issues and potential risks when using computers.

Cambridge O Level Computer Science is an ideal foundation for further study in Computer Science. Understanding the principles of Computer Science provides learners with the underpinning knowledge required for many other subjects in science and engineering, and the skills learnt can also be used in everyday life.

Sections	Topics
Section 1 **Theory of Computer Science**	1.1 Data representation 1.1.1 Binary systems 1.1.2 Hexadecimal 1.1.3 Data storage 1.2 Communication and Internet technologies 1.2.1 Data transmission 1.2.2 Security aspects 1.2.3 Internet principles of operation 1.3 Hardware and software 1.3.1 Logic gates 1.3.2 Computer architecture and the fetch-execute cycle 1.3.3 Input devices 1.3.4 Output devices 1.3.5 Memory, storage devices and media 1.3.6 Operating systems 1.3.7 High- and low-level languages and their translators 1.4 Security 1.5 Ethics
Section 2 **Practical Problem-solving and Programming**	2.1 Algorithm design and problem-solving 2.1.1 Problem-solving and design 2.1.2 Pseudocode and flowcharts 2.2 Programming 2.2.1 Programming concepts 2.2.2 Data structures; arrays 2.3 Databases

Assessment at a glance

For Cambridge O Level Computer Science, candidates take two components: Paper 1 and Paper 2.

Component

Paper 1 Theory This written paper contains short-answer and structured questions. All questions are compulsory. No calculators are permitted in this paper. 75 marks Externally assessed.	1 hour 45 minutes	60%
Paper 2 Problem-solving and Programming This written paper contains short-answer and structured questions. All questions are compulsory. 20 of the marks for this paper are from questions set on the pre‑release material.[1] No calculators are permitted in this paper. 50 marks Externally assessed.	1 hour 45 minutes	40%

Availability

This syllabus is examined in the June and November examination series.
This syllabus is available to private candidates.
Detailed timetables are available from www.cie.org.uk/examsofficers
Centres in the UK that receive government funding are advised to consult the Cambridge website www.cie.org.uk for the latest information before beginning to teach this syllabus.

Combining this with other syllabuses

Candidates can combine this syllabus in an examination series with any other Cambridge syllabus, except:
syllabuses with the same title at the same level
0478 Cambridge IGCSEl Computer Science
Please note that Cambridge O Level, Cambridge IGCSE and Cambridge International Level 1/Level 2 Certificate syllabuses are at the same level.
1 The pre-release material for Paper 2 will be made available to Centres the January before the June examination, and the July before the November examination. It will also be reproduced in the question paper. Candidates are not permitted to bring any prepared material into the examination.

Syllabus aims and assessment objectives

Syllabus aims

Cambridge O Level Computer Science syllabus aims are to develop:

- computational thinking that is thinking about what can be computed and how, and includes consideration of the data required
- understanding of the main principles of solving problems by using computers
- understanding that every computer system is made up of sub-systems, which in turn consist of further sub-systems
- understanding of the component parts of computer systems and how they interrelate, including software, data, hardware, communications and people
- Skills necessary to apply understanding to solve computer-based problems using a high-level programming language.

Assessment objectives

AO1 Recall, select and communicate knowledge and understanding of computer technology. AO2 Apply knowledge, understanding and skills to solve computing or programming problems. **AO3** Analyse, evaluate, make reasoned judgements and present conclusions.

Relationship between assessment objectives and components

The approximate weightings allocated to each of the assessment objectives are summarised below.

Assessment objective	Paper 1	Paper 2	Weighting for qualification
AO1	32%	8%	40%
AO2	16%	24%	40%
AO3	12%	8%	20%
Total	60%	40%	100%

Syllabus content

For Cambridge O Level Computer Science, the assessment is by written examination but the learning should happen in a mainly practical way: problem-solving and programming

Section 1 Theory of Computer Science

1.1 Data representation: Candidates should be able to:

1.1.1 Binary systems

- Recognise the use of binary numbers in computer systems
- convert positive denary integers into binary and positive binary integers into denary (a maximum of 16 bits will be used)
- show understanding of the concept of a byte and how the byte is used to measure memory size
- use binary in computer registers for a given application (such as in robotics, digital instruments and counting systems)

1.1.2 Hexadecimal

- represent positive numbers in hexadecimal notation
- show understanding of the reasons for choosing hexadecimal notation to represent numbers
- convert positive hexadecimal integers to and from denary (a maximum of four hexadecimal digits will be required)
- convert positive hexadecimal integers to and from binary (a maximum of 16 bit binary numbers will be required)
- represent numbers stored in registers and main memory as hexadecimal
- identify current uses of hexadecimal numbers in computing, such as defining colours in Hypertext Markup Language (HTML), Media Access Control (MAC) addresses, assembly languages and machine code, debugging

1.1.3 Data storage

- show understanding that sound (music), pictures, video, text and numbers are stored in different formats
- identify and describe methods of error detection and correction, such as parity checks, check digits, checksums and Automatic Repeat reQuests (ARQ)
- show understanding of the concept of Musical Instrument Digital Interface (MIDI) files, JPEG files, MP3 and MP4 files
- show understanding of the principles of data compression (lossless and lossy) applied to music/ video, photos and text files

1.2 Communication and Internet technologies

Candidates should be able to:

1.2.1 Data transmission

- show understanding of what is meant by transmission of data
- distinguish between serial and parallel data transmission
- distinguish between simplex, duplex and half-duplex data transmission
- show understanding of the reasons for choosing serial or parallel data transmission
- show understanding of the need to check for errors

- explain how parity bits are used for error detection
- show understanding of the use of serial and parallel data transmission, in Universal Serial Bus (USB) and Integrated Circuit (IC)

1.2.2 Security aspects

(This section links with section 1.4 of the syllabus.)

- show understanding of the security aspects of using the Internet and understand what methods are available to help minimise the risks
- show understanding of the Internet risks associated with malware, including viruses, spyware and hacking
- explain how anti-virus and other protection software helps to protect the user from security risks

1.2.3 Internet principles of operation

- show understanding of the role of the browser
- show understanding of the role of an Internet Service Provider (ISP)
- show understanding of what is meant by hypertext transfer protocol (http and https) and HTML
- distinguish between HTML structure and presentation
- show understanding of the concepts of MAC address, Internet Protocol (IP) address, Uniform Resource Locator (URL) and cookies

1.3 Hardware and software

Candidates should be able to:

1.3.1 Logic gates

- use logic gates to create electronic circuits
- understand and define the functions of NOT, AND, OR, NAND, NOR and XOR (EOR) gates, including the binary output produced from all the possible binary inputs (all gates, except the NOT gate, will have 2 inputs only)
- draw truth tables and recognise a logic gate from its truth table
- recognise and use the following standard symbols used to represent logic gates:
  produce truth tables for given logic circuits, for example

A	B	C	Output
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

  produce a logic circuit to solve a given problem or to implement a given written logic statement

1.3.2 Computer architecture and the fetch-execute cycle

- show understanding of the basic Von Neumann model for a computer system and the stored program concept (program instructions and data are stored in main memory and instructions are fetched and executed one after another)
- describe the stages of the fetch-execute cycle, including the use of registers and buses

1.3.3 Input devices

- describe the principles of operation (how each device works) of these input devices: 2D and 3D scanners, barcode readers, Quick Response (QR) code readers, digital cameras, keyboards, mice, touch screens, interactive whiteboard, microphones
- describe how these principles are applied to real-life scenarios, for example: scanning of passports at airports, barcode readers at supermarket checkouts, and touch screens on mobile devices
- describe how a range of sensors can be used to input data into a computer system, including light, temperature, magnetic field, gas, pressure, moisture, humidity, pH and motion
- describe how these sensors are used in real-life scenarios, for example: street lights, security devices, pollution control, games, and household and industrial applications

1.3.4 Output devices

- describe the principles of operation of the following output devices: inkjet, laser and 3D printers; 2D and 3D cutters; speakers and headphones; actuators; flat-panel display screens, such as Liquid Crystal Display (LCD) and Light-Emitting Diodes (LED) display; LCD projectors and Digital Light Projectors (DLP)
- describe how these principles are applied to real-life scenarios, for example: printing single

items on demand or in large volumes; use of small screens on mobile devices
1.3.5 Memory, storage devices and media • show understanding of the difference between: primary, secondary and off-line storage and provide examples of each, such as: primary: Read Only Memory (ROM), and Random Access Memory (RAM) secondary: hard disk drive (HDD) and Solid State Drive (SSD); off-line: Digital Versatile Disc (DVD), Compact Disc (CD), Blu-ray disc, USB flash memory and removable HDD • describe the principles of operation of a range of types of storage device and media including magnetic, optical and solid state • describe how these principles are applied to currently available storage solutions, such as SSDs, HDDs, USB flash memory, DVDs, CDs and Blu-ray discs • calculate the storage requirement of a file
1.3.6 Operating systems • describe the purpose of an operating system (Candidates will be required to understand the purpose and function of an operating system and why it is needed. They will not be required to understand how operating systems work.) • show understanding of the need for interrupts
1.3.7 High- and low-level languages and their translators • show understanding of the need for both high-level and low-level languages • show understanding of the need for compilers when translating programs written in a high-level language • show understanding of the use of interpreters with high-level language programs • show understanding of the need for assemblers when translating programs written in assembly language

1.4 Security

Candidates should be able to:

1.4.1 • show understanding of the need to keep data safe from accidental damage, including corruption and human errors • show understanding of the need to keep data safe from malicious actions, including unauthorised viewing, deleting, copying and corruption
1.4.2 • show understanding of how data are kept safe when stored and transmitted, including: — use of passwords, both entered at a keyboard and biometric — use of firewalls, both software and hardware, including proxy servers — use of security protocols such as Secure Socket Layer (SSL) and Transport Layer Security (TLS) —use of symmetric encryption (plain text, cypher text and use of a key) showing understanding that increasing the length of a key increases the strength of the encryption
1.4.3 • show understanding of the need to keep online systems safe from attacks including denial of service

attacks, phishing, pharming
1.4.4 • describe how the knowledge from 1.4.1, 1.4.2 and 1.4.3 can be applied to real-life scenarios including, for example, online banking, shopping

1.5 Ethics

Candidates should be able to:

• show understanding of computer ethics, including copyright issues and plagiarism • distinguish between free software, freeware and shareware • show understanding of the ethical issues raised by the spread of electronic communication and computer systems, including hacking, cracking and production of malware

Section 2 Practical Problem-solving and Programming

2.1 Algorithm design and problem-solving

Candidates should be able to:

2.1.1 Problem-solving and design

- show understanding that every computer system is made up of sub-systems, which in turn are made up of further sub-systems
- use top-down design, structure diagrams, flowcharts, pseudocode, library routines and sub-routines
- work out the purpose of a given algorithm
- explain standard methods of solution
- suggest and apply suitable test data
- understand the need for validation and verification checks to be made on input data (validation could include range checks, length checks, type checks and check digits)
- use trace tables to find the value of variables at each step in an algorithm
- identify errors in given algorithms and suggest ways of removing these errors
- produce an algorithm for a given problem (either in the form of pseudocode or flowchart) comment on the effectiveness of a given solution

2.1.2 Pseudocode and flowcharts

- understand and use pseudocode for assignment, using ←
- understand and use pseudocode, using the following conditional statements:
  IF … THEN … ELSE … ENDIF
  CASE … OF … OTHERWISE … ENDCASE
- understand and use pseudocode, using the following loop structures:
  FOR … TO … NEXT
  REPEAT … UNTIL
  WHILE … DO … ENDWHILE
- understand and use pseudocode, using the following commands and statements:
  INPUT and OUTPUT (e.g.
  READ and PRINT) totalling
  (e.g. Sum ← Sum + Number)
  counting (e.g. Count ← Count
  + 1)
- understand and use standard flowchart symbols to represent the above statements, commands and structures

(Candidates are advised to try out solutions to a variety of different problems on a computer using a language of their choice; no particular programming language will be assumed in this syllabus.)

2.2 Programming

Candidates should be able to:

2.2.1 Programming concepts

- declare and use variables and constants
- understand and use basic data types: Integer, Real, Char, String and Boolean
- understand and use the concepts of sequence, selection, repetition, totalling and counting

use predefined procedures/functions

2.2.2 Data structures; arrays

- declare and use one-dimensional arrays, for example: A[1:n]
- show understanding of the use of one-dimensional arrays, including the use of a variable as an index in an array read or write values in an array using a FOR … TO … NEXT loop

2.3 Databases

Candidates should be able to:

- define a single-table database from given data storage requirements
- choose and specify suitable data types
- choose a suitable primary key for a database table
- perform a query-by-example from given search criteria

Scheme of assessment

Paper 1 Theory

This is a compulsory question paper, consisting of short-answer and structured questions set on Section 1 of the syllabus content. All questions are compulsory. Candidates will answer on the question paper.

Paper 2 Problem-solving and Programming

This is a compulsory question paper, consisting of short-answer and structured questions set on Section 2 of the syllabus content. All questions are compulsory. Candidates will answer on the question paper.

20 of the marks in this paper are from questions set on tasks provided in the Paper 2 Problem-solving and Programming pre-release material.

Centres need to be aware that in order to prepare best their candidates for this paper, they should plan for sufficient practical sessions within their lesson timetable and teach the contents of the section in a largely practical way. Candidates will be expected to be able to program in a high-level programming language to be chosen by the Centre. The programming language should be procedural.

There will be some examining of knowledge with understanding, but most of the credit will be for using techniques and skills to solve problems. The examination questions will require candidates to have practical programming experience, including writing their own programs, executing (running), testing and debugging them. Knowledge of programming language syntax will not be examined; in all cases the logic will be more important than the syntax.

Paper 2 Problem-solving and Programming pre-release material

The Paper 2 Problem-solving and Programming pre-release material will be made available to Centres the January before the June examination, and the July before the November examination. It will also be reproduced in the question paper. Candidates are not permitted to bring any prepared material into the examination.

Centres are advised to encourage their candidates to develop solutions to tasks using a high-level programming language, such as Visual Basic, Pascal/Delphi or Python. The purpose of the pre-release material tasks is to direct candidates to some of the topics which will be examined in Paper 2. Teachers are expected to incorporate these tasks into their lessons and give support in finding methods and reaching solutions. 20 of the marks in this paper will be from questions testing candidates' understanding gained from developing programmed solutions to these tasks.