

SYLLABUS

**Cambridge International AS and A Level
Computing**

9691

For examination in June 2016

Changes to syllabus for 2016

This syllabus has been updated, but there are no significant changes.

Cambridge International Examinations retains the copyright on all its publications. Registered Centres are permitted to copy material from this booklet for their own internal use. However, we cannot give permission to Centres to photocopy any material that is acknowledged to a third party even for internal use within a Centre.

® IGCSE is the registered trademark of Cambridge International Examinations

© Cambridge International Examinations 2014

Contents

| | |
|--|----|
| 1. Introduction | 2 |
| 1.1 Why choose Cambridge? | |
| 1.2 Why choose Cambridge International AS and A Level? | |
| 1.3 Why choose Cambridge International AS and A Level Computing? | |
| 1.4 Cambridge AICE (Advanced International Certificate of Education) Diploma | |
| 1.5 How can I find out more? | |
| 2. Teacher support..... | 6 |
| 2.1 Support materials | |
| 2.2 Resource lists | |
| 2.3 Training | |
| 3. Assessment at a glance | 7 |
| 4. Syllabus aims and assessment objectives | 9 |
| 4.1 Syllabus aims | |
| 4.2 Assessment objectives | |
| 4.3 Weighting of assessment objectives | |
| 5. Description of components | 11 |
| 5.1 Paper 1 | |
| 5.2 Paper 2 | |
| 5.3 Paper 3 | |
| 5.4 Paper 4 | |
| 6. Syllabus content | 12 |
| 7. Coursework | 35 |
| 8. Appendix..... | 37 |
| 8.1 Guidance on selecting the Computing Project | |
| 8.2 Guidance on marking the Computing Project | |
| Paper 4: Marking Grid (attach to Project) | |
| 9. Other information | 50 |

1. Introduction

1.1 Why choose Cambridge?

Recognition

Cambridge International Examinations is the world's largest provider of international education programmes and qualifications for learners aged 5 to 19. We are part of Cambridge Assessment, a department of the University of Cambridge, trusted for excellence in education. Our qualifications are recognised by the world's universities and employers.

Cambridge International AS and A Levels are recognised around the world by schools, universities and employers. The qualifications are accepted as proof of academic ability for entry to universities worldwide, though some courses do require specific subjects.

Cambridge International A Levels typically take two years to complete and offer a flexible course of study that gives learners the freedom to select subjects that are right for them.

Cambridge International AS Levels often represent the first half of an A Level course but may also be taken as a freestanding qualification. The content and difficulty of a Cambridge International AS Level examination is equivalent to the first half of a corresponding Cambridge International A Level. Cambridge AS Levels are accepted in all UK universities and carry half the weighting of an A Level. University course credit and advanced standing is often available for Cambridge International AS and A Levels in countries such as the USA and Canada.

Learn more at www.cie.org.uk/recognition

Excellence in education

Our mission is to deliver world-class international education through the provision of high-quality curricula, assessment and services.

More than 9000 schools are part of our Cambridge learning community. We support teachers in over 160 countries who offer their learners an international education based on our curricula and leading to our qualifications. Every year, thousands of learners use Cambridge qualifications to gain places at universities around the world.

Our syllabuses are reviewed and updated regularly so that they reflect the latest thinking of international experts and practitioners and take account of the different national contexts in which they are taught.

Cambridge programmes and qualifications are designed to support learners in becoming:

- **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.

Support for teachers

A wide range of support materials and resources is available for teachers and learners in Cambridge schools. Resources suit a variety of teaching methods in different international contexts. Through subject discussion forums and training, teachers can access the expert advice they need for teaching our qualifications. More details can be found in Section 2 of this syllabus and at www.cie.org.uk/teachers

Support for exams officers

Exams officers can trust in reliable, efficient administration of exam entries and excellent personal support from our customer services. Learn more at www.cie.org.uk/examsOfficers

Not-for-profit, part of the University of Cambridge

We are a not-for-profit organisation where the needs of the teachers and learners are at the core of what we do. We continually invest in educational research and respond to feedback from our customers in order to improve our qualifications, products and services.

Our systems for managing the provision of international qualifications and education programmes for learners aged 5 to 19 are certified as meeting the internationally recognised standard for quality management, ISO 9001:2008. Learn more at www.cie.org.uk/ISO9001

1.2 Why choose Cambridge International AS and A Level?

Cambridge International AS and A Levels are international in outlook, but retain a local relevance. The syllabuses provide opportunities for contextualised learning and the content has been created to suit a wide variety of schools, avoid cultural bias and develop essential lifelong skills, including creative thinking and problem-solving.

Our aim is to balance knowledge, understanding and skills in our programmes and qualifications to enable candidates to become effective learners and to provide a solid foundation for their continuing educational journey. Cambridge International AS and A Levels give learners building blocks for an individualised curriculum that develops their knowledge, understanding and skills.

Schools can offer almost any combination of 60 subjects, and learners can specialise or study a range of subjects, ensuring a breadth of knowledge. Giving learners the power to choose helps motivate them throughout their studies.

Through our professional development courses and our support materials for Cambridge International AS and A 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 International AS and A Levels have a proven reputation for preparing learners well for university, employment and life. They help develop the in-depth subject knowledge and understanding which are so important to universities and employers.

Learners studying Cambridge International AS and A Levels have the opportunities to:

- acquire an in-depth subject knowledge
- develop independent thinking skills
- apply knowledge and understanding to new as well as familiar situations
- handle and evaluate different types of information sources
- think logically and present ordered and coherent arguments
- make judgements, recommendations and decisions
- present reasoned explanations, understand implications and communicate them clearly and logically
- work and communicate in English.

Guided learning hours

Cambridge International A Level syllabuses are designed on the assumption that learners have about 360 guided learning hours per subject over the duration of the course. Cambridge International AS Level syllabuses are designed on the assumption that learners have about 180 guided learning hours per subject over the duration of the course. This is for guidance only and 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 International AS and A Level Computing?

Cambridge International A Level and AS Level Computing are accepted by universities and employers as proof of essential knowledge and ability.

This syllabus is designed to give greater flexibility both to teachers and to candidates. It is envisaged that candidates will use the skills and knowledge of computing acquired through this course in one of three ways:

- to provide a general understanding and perspective of the use of computer technology and systems, which will inform their decisions and support their participation in an increasingly technologically dependent society
- to provide the necessary skills and knowledge to seek employment in areas that utilise computing
- to develop their knowledge and understanding of computing through entry to higher education, where this qualification will provide a useful foundation for further study of computing or more specialist aspects of computing.

Prior learning

Candidates beginning this course are not expected to have studied Computing or ICT previously.

Progression

Cambridge International A Level Computing provides a suitable foundation for the study of Computing or related courses in higher education. Equally it is suitable for candidates intending to pursue careers or further study in Computing or ICT, or as part of a course of general education.

Cambridge International AS Level Computing constitutes the first half of the Cambridge International A Level course in Computing and therefore provides a suitable foundation for the study of Computing at Cambridge International A Level and thence for related courses in higher education. Depending on local university

entrance requirements, it may permit or assist progression directly to university courses in Computing or some other subjects. It is also suitable for candidates intending to pursue careers or further study in Computing or ICT, or as part of a course of general education.

1.4 Cambridge AICE (Advanced International Certificate of Education) Diploma

Cambridge AICE Diploma is the group award of the Cambridge International AS and A Level. It gives schools the opportunity to benefit from offering a broad and balanced curriculum by recognising the achievements of candidates who pass examinations in three different curriculum groups:

- Mathematics and Science (Group 1)
- Languages (Group 2)
- Arts and Humanities (Group 3)

A Cambridge International A Level counts as a double-credit qualification and a Cambridge International AS Level counts as a single-credit qualification within the Cambridge AICE Diploma award framework.

To be considered for an AICE Diploma, a candidate must earn the equivalent of six credits by passing a combination of examinations at either double credit or single credit, with at least one course coming from each of the three curriculum groups.

Computing (9691) is in Group 1, Mathematics and Science.

Credits gained from Cambridge AS Level Global Perspectives (8987) or Cambridge Pre-U Global Perspectives and Independent Research (9766) can be counted towards the Cambridge AICE Diploma, but candidates must also gain at least one credit from each of the three curriculum groups to be eligible for the award.

Learn more about the Cambridge AICE Diploma at www.cie.org.uk/qualifications/academic/uppersec/aice

The Cambridge AICE Diploma is awarded from examinations administered in the June and November series each year.

1.5 How can I find out more?

If you are already a Cambridge school

You can make entries for this qualification through your usual channels. If you have any questions, please contact us at info@cie.org.uk

If you are not yet a Cambridge school

Learn about the benefits of becoming a Cambridge school at www.cie.org.uk/startcambridge. Email us at info@cie.org.uk to find out how your organisation can register to become a Cambridge school.

2. Teacher support

2.1 Support materials

Cambridge syllabuses, past question papers and examiner reports to cover the last examination series are on the *Syllabus and Support Materials* DVD, which we send to all Cambridge schools.

You can also go to our public website at www.cie.org.uk/alevel to download current and future syllabuses together with specimen papers or past question papers and examiner reports from one series.

For teachers at registered Cambridge schools a range of additional support materials for specific syllabuses is available from Teacher Support, our secure online support for Cambridge teachers. Go to <http://teachers.cie.org.uk> (username and password required).

2.2 Resource lists

We work with publishers providing a range of resources for our syllabuses including textbooks, websites, CDs, etc. Any endorsed, recommended and suggested resources are listed on both our public website and on Teacher Support.

The resource lists can be filtered to show all resources or just those which are endorsed or recommended by Cambridge. Resources endorsed by Cambridge go through a detailed quality assurance process and are written to align closely with the Cambridge syllabus they support.

2.3 Training

We offer a range of support activities for teachers to ensure they have the relevant knowledge and skills to deliver our qualifications. See www.cie.org.uk/events for further information.

3. Assessment at a glance

Centres and candidates may choose:

- to take components 1, 2, 3 and 4 in the same examination series, leading to the full Cambridge International A Level
- to follow a **staged** assessment route by taking papers 1 and 2 (for the AS qualification) in one series, then papers 3 and 4 (for the full Cambridge International A Level) at a later series
- to take papers 1 and 2 only (for the AS qualification).

| Paper | Marks | Weighting (%) | | |
|--|-------|---------------|----|----|
| | | AS | A2 | A |
| Paper 1 1½ hours Written paper on Section 1 of syllabus No calculators allowed. | 75 | 50 | – | 25 |
| Paper 2 2 hours Written paper on Section 2 of syllabus | 75 | 50 | – | 25 |
| Paper 3 2 hours Written paper on Section 3 of syllabus, also assuming knowledge from Section 1 No calculators allowed. | 90 | – | 60 | 30 |
| Paper 4 Computing project | 60 | – | 40 | 20 |

Advanced Subsidiary (AS) forms 50% of the assessment weighting of the full Advanced Level.

Availability

This syllabus is examined in the June examination series.

The Cambridge International AS Level syllabus is available to private candidates. The Cambridge International A Level syllabus is **not** 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.

4. Syllabus aims and assessment objectives

4.1 Syllabus aims

The aims of a course based on this syllabus, whether leading to an AS or A Level qualification are:

- to develop an understanding of the main principles of solving problems using computers
- to develop an understanding of the range of applications of computers and the effects of their use
- to develop an understanding of the organisation of computer systems including software, data, hardware, communications and people
- to acquire the skills necessary to apply this understanding to developing computer-based solutions to problems

An additional aim for a course leading to the full Cambridge International A Level qualification is:

- to develop an understanding of the main principles of systems analysis and design, methods of problem formulation and planning of solutions using computers, and systematic methods of implementation, testing and documentation.

4.2 Assessment objectives

The assessment objectives below are for both the AS and A2 qualifications.

A. Knowledge with understanding

Candidates should be able to:

- describe and explain the impact of computing in a range of applications, and show an understanding of the characteristics of computer systems (hardware, software and communication) which allow effective solutions to be achieved
- describe and explain the need for and the use of various forms of data organisation and processing to support the information requirements of a particular application
- describe and explain the systematic development of high quality solutions to problems and the techniques appropriate for implementing such solutions
- comment critically on the social, legal, ethical and other consequences of the use of computers.

B. Skills

Candidates should be able to:

- analyse a problem and identify the parts which are appropriate for a computer-based solution
- select, justify and apply appropriate techniques and principles to develop data structures and algorithms for the solution of problems
- design, implement and document an effective solution using appropriate hardware, software and programming languages.

4.3 Weighting of assessment objectives

| Paper | Percentage of Advanced Level | | |
|----------|------------------------------|--------|---------------|
| | Knowledge with understanding | Skills | Weighting (%) |
| 1 | 20 ± 2 | 5 ± 2 | 25 |
| 2 | 8 ± 2 | 17 ± 2 | 25 |
| 3 | 15 ± 2 | 15 ± 2 | 30 |
| 4 | 5 ± 2 | 15 ± 2 | 20 |

5. Description of components

5.1 Paper 1

This paper will consist of a variable number of compulsory questions of variable mark value. Candidates will answer on the question paper.

This paper will be set according to the content of Section 1 of the syllabus.

No calculators allowed.

5.2 Paper 2

Candidates will answer on the question paper.

The paper will consist of a number (between 3 and 5) of compulsory questions, each of which will be based around a scenario and which will develop a theme through a number of graded parts to the question. The emphasis will be based around the need to demonstrate skills in using techniques that have been learned through study of the syllabus. There will be some examining of knowledge and understanding but the bulk of the credit will be for using those techniques and that knowledge to solve problems.

Candidates will be expected to be able to program in a language to be chosen by the Centre but the advice is that the language chosen should be procedural. In all cases the logic will be of more importance than the syntax.

The question content will be based on Section 2 of the syllabus. This will mean that a certain understanding of the content of Section 1 will be expected, but there will be no questions which are aimed specifically at testing Section 1.

Paper 2 is an examination which will be externally marked. Centres should ensure that candidates are taught the content of this module in a largely practical way in order to prepare them for the requirements of the project in module 4. Centres are reminded that the content of the examination will involve the assessment of the skills and techniques rather than any major problem solution.

5.3 Paper 3

This paper will consist of a variable number of compulsory questions of variable mark value. Candidates will answer on the question paper.

This paper will be set according to the content of Section 3 of the syllabus, but will also assume knowledge learned in Section 1 and 2.

No calculators allowed.

5.4 Paper 4

Further details of the Computing Project can be found in Section 6: Coursework.

6. Syllabus content

This syllabus is set out in the form of teaching sections. Each teaching section is assessed by its associated paper. The Advanced Subsidiary syllabus consists of teaching Sections 1 and 2 only, and the Advanced Level syllabus consists of all four teaching sections.

The subject content for each section is shown below.

| Syllabus section | Paper | Section title |
|------------------|-------|---|
| 1 | 1 | Computer systems, communications and software |
| 2 | 2 | Practical programming techniques |
| 3 | 3 | System software mechanisms, machine architecture, database theory, programming paradigms and integrated information systems |
| 4 | 4 | Computing Project |

Each section is presented as a set of sub-sections, each with details of content and associated learning outcomes. An indication of recommended prior knowledge is given for each section, together with details of any links to other sections.

Section 1: Computer systems, communications and software

Section 1 is the foundation for all subsequent sections. It provides candidates with an understanding of the core aspects of computer systems, which is developed and enhanced in subsequent sections.

Section 2: Practical programming techniques

Section 2 requires candidates to familiarise themselves with the techniques necessary to solve problems using a computer and specifically using programming to implement algorithmic solutions. To that end the topics covered will be:

- the need to design a solution before attempting to implement it
- procedural programming techniques which include the basic constructs of sequence, selection and iteration; the importance and use of data types and data structures
- common syntax and key instructions of procedural languages
- the application of these techniques in program writing and the testing and running of these solutions.

Section 3: System software mechanisms, machine architecture, database theory, programming paradigms and integrated information systems

Section 3 provides candidates with further opportunity to extend the skills, knowledge and understanding of computing concepts gained in Section 1, to a range of applications in which computer systems are used.

Section 4: Computing Project

Section 4 requires candidates to identify a well-defined user-driven problem, involving a third-party user, and to generate a solution.

This is done using software tools chosen by the candidate and should include a programming language, an appropriate applications package or other software. Work on the project should begin in parallel with work on Section 3.

Section 1: Computer systems, communications and software

This section provides candidates with an understanding of the following core aspects of computer systems:

- components of a computer system and modes of use
- system software
- data: its representation, structure and management
- hardware
- data transmission and networking.

The system life cycle is studied with reference to particular applications, so candidates are expected to look at a range of different types of application areas. Although candidates are not expected to have specific knowledge of every one of these, they should be able to make use of relevant examples for the purpose of illustration.

This section also provides candidates with understanding of the following aspects of computer systems:

- system life cycle
- choosing appropriate application software
- handling of data in information systems
- designing the user interface
- logic gates.

1.1 Components of a computer system and modes of use

Content

1.1.1 Types of hardware

1.1.2 Types of software

Learning outcomes

Candidates should be able to:

- define the terms hardware, software, input device, storage device and output device
- describe the purpose of input devices, storage devices and output devices
- define the different types of software: operating system and generic/common application software

1.2 System software

Content

- 1.2.1 Operating systems
- 1.2.2 User interfaces
- 1.2.3 Utility software

Learning outcomes

Candidates should be able to:

- (a) describe the purpose of operating systems
- (b) describe the characteristics of different types of operating systems and their uses: batch, real-time (transaction processing and process control), single-user, multi-user, multi-programming, multi-tasking and network
- (c) identify a range of applications requiring batch processing, transaction processing and process control
- (d) describe different types of user interface: forms, menus, GUI, natural language and command-line, suggesting the characteristics of user interfaces which make them appropriate for use by different types of user
- (e) describe the purpose of a range of utility software, e.g. disk formatting, file handling, hardware drivers, file compression and virus checkers

1.3 Data: its representation, structure and management

Content

- 1.3.1 Data types
- 1.3.2 Data structures
- 1.3.3 Data management

Learning outcomes

Candidates should be able to:

- (a) explain the use of codes to represent a character set (e.g. ASCII and Unicode)
- (b) explain the representation of different data types: integer, Boolean, date/time, currency, character and string
- (c) express positive integers in binary form
- (d) understand the structure of arrays (one- and two-dimensional), including initialising arrays, reading data into arrays and performing a simple serial search on an array
- (e) describe the LIFO and FIFO features of stacks and queues
- (f) explain how data is stored in files in the form of fixed-length records comprising items in fields
- (g) define and explain the difference between serial, sequential, indexed sequential and random access to data, using examples and stating their comparative advantages and disadvantages
- (h) describe how serial, sequential and random organisation and access to files may be implemented using indexes and hashing as appropriate
- (i) select appropriate data types/data structures for a given problem and explain the advantages and disadvantages of alternative choices
- (j) explain the procedures involved in backing up data and archiving, including the reasons for backing up and archiving

1.4 Hardware

Content

- 1.4.1 Processor components
- 1.4.2 Primary and secondary storage
- 1.4.3 Peripheral devices

Learning outcomes

Candidates should be able to:

- (a) describe the function and purpose of the control unit, memory unit and arithmetic logic unit (ALU) as individual parts of a processor
- (b) explain the difference between types of primary memory and their uses (RAM, ROM)
- (c) describe the basic features, advantages, disadvantages and use of secondary storage media, e.g. magnetic, optical and solid state
- (d) describe use of buffers and interrupts in the transfer of data between peripheral devices and primary memory
- (e) describe a range of common peripheral devices in terms of their features, benefits, drawbacks and uses
- (f) relate the choice of peripheral device to a given application, justifying the choices made
- (g) understand the potential problem of speed mismatch between peripheral and processor

1.5 Data transmission and networking

Content

- 1.5.1 Data transmission
- 1.5.2 Circuit switching and packet switching
- 1.5.3 Protocols
- 1.5.4 Networking

Learning outcomes

Candidates should be able to:

- (a) describe the characteristics of a local area network (LAN) and a wide area network (WAN)
- (b) show an understanding of the hardware and software needed for a local area network (LAN) and for accessing a wide area network (WAN)
- (c) describe basic network topologies (bus, star and ring) explaining the benefits and drawbacks of each topology and typical applications where each topology would be used
- (d) describe the different types of data transmission: serial and parallel; simplex, half duplex and full duplex modes
- (e) explain the relationship between baseband and broadband
- (f) recognise that errors can occur in data transmission; explain the use of parity checks, echoing and checksums in detecting and correcting these errors, and the use of parity blocks to aid self-checking
- (g) explain the difference between packet switching and circuit switching
- (h) define the term protocol
- (i) describe the need for communication between devices, and between computers, and explain the need for protocols to establish communication links
- (j) explain the need for both physical and logical protocols

1.6 System life cycle

Content

- 1.6.1 Identification of problem
- 1.6.2 Feasibility study
- 1.6.3 Information collection
- 1.6.4 Analysis of a problem, based upon information collected, including producing a requirements specification
- 1.6.5 Design of system to fit requirements
- 1.6.6 Development and testing of system
- 1.6.7 Installation of system
- 1.6.8 Maintenance of system
- 1.6.9 Obsolescence

Learning outcomes

Candidates should, with reference to particular applications, be able to:

- (a) explain the importance of defining a problem accurately
- (b) describe the function and purpose of a feasibility study
- (c) explain the importance of determining the information requirements of a system and describe different methods of fact finding, highlighting the advantages and disadvantages of each method
- (d) describe what is involved when analysing the requirements of a system, explaining the nature of the requirements specification and its content, identifying inefficiencies/problems, user requirements and hardware and software requirements
- (e) design the data structures, inputs, outputs and processing using diagrammatic representations where appropriate (including the use of dataflow diagrams (DFDs) and system flowcharts)
- (f) explain the importance of evaluating the system against initial specifications
- (g) explain the content and importance of documentation in the system life cycle, including the requirements specification, design specification, program specification and documentation
- (h) explain the importance of testing and installation planning, including the method of installation
- (i) explain the reasons for maintaining the system

1.7 Choosing appropriate applications software

Content

- 1.7.1 Custom-written software versus off-the-shelf software packages
- 1.7.2 Application software

Learning outcomes

Candidates should, within context, be able to:

- (a) distinguish between custom-written software and off-the-shelf software packages, and discuss the benefits and drawbacks of each in given situations
- (b) identify the features of common applications found in business, commercial and industrial applications, e.g. stock control, payroll, process control, point-of-sale systems
- (c) identify suitable common generic application software for particular application areas, e.g. word processor, spreadsheet, desktop publisher (DTP), presentation software, graphics packages (bit mapped and vector graphics), and justify the choices
- (d) identify application areas for which generic application software is not appropriate
- (e) describe the purpose and impact of different types of generic application software, e.g. word processor, spreadsheet, desktop publisher (DTP), presentation software, graphics packages

1.8 Handling of data in information systems

Content

- 1.8.1 Data capture, preparation and data input
- 1.8.2 Validation and verification of data
- 1.8.3 Outputs from a system
- 1.8.4 Knowledge-based systems

Learning outcomes

Candidates should, within a context, be able to:

- (a) describe manual and automated methods of capturing and inputting data into a system, including form design, keyboard entry, barcodes, Optical Mark Recognition (OMR), magnetic stripe cards, Optical Character Recognition (OCR), sensors and data logging, touch screens, chip and pin
- (b) describe image capture by use of a scanner, video capture card and digital camera/camcorder
- (c) explain the techniques of validation and verification, and describe validation tests which can be carried out on data
- (d) describe possible output formats such as graphs, reports, interactive presentations, sound, video, images and animations stating the benefits and drawbacks of each format
- (e) discuss the need for a variety of output formats according to the target audience
- (f) describe knowledge-based (expert) systems, how they are created and how they function
- (g) explain the use of knowledge-based (expert) systems as a diagnostic tool

1.9 Designing the user interface

Content

1.9.1 Interface design

Learning outcomes

Candidates should be able to:

- (a) discuss the importance of good interface design
- (b) discuss human computer interaction (HCI) design issues such as the use of colour, layout, and content
- (c) identify the required characteristics of a user interface with respect to information, type of interface, type of user, type of application, physical location and current technology

1.10 Logic gates

Content

1.10.1 Uses of logic gates to translate Boolean concepts into physical uses

Learning outcomes

Candidates should, within a context, be able to:

- (a) understand and define the function of AND, OR, NOT, NAND and NOR logic gates including the binary output produced from all possible binary inputs
- (b) calculate the outcome from a circuit of logic gates given the input by producing truth tables for given logic circuits
- (c) produce a simple logic circuit from a given written statement (e.g. if A AND B are on AND if C is on then the lights will be on)

Section 2: Practical programming techniques

This section provides candidates with an understanding of the techniques required for programming through a study of the following topics:

- designing solutions to problems
- the structure of procedural programs
- data types and data structures
- common facilities of procedural languages
- writing maintainable programs
- testing and running a solution.

2.1 Designing solutions to problems

Content

2.1.1 Design of the input, output and interface

2.1.2 Use of structure diagrams to describe the modular nature of a solution

2.1.3 Use of program flowcharts and pseudocode to describe the steps of an algorithm

Learning outcomes

Candidates should be able to:

- discuss the importance of good interface design
- design and document data capture forms, screen layouts, report layouts or other forms of input and output (e.g. sound) for a given problem
- explain the benefits of designing a solution to a problem by splitting it up into smaller problems (top-down/modular design)
- produce and describe top-down/modular designs using appropriate techniques, including structure diagrams, showing stepwise refinement
- produce algorithms to solve problems using both a program flowchart and pseudocode
- understand algorithms presented in the form of program flowcharts and pseudocode

2.2 The structure of procedural programs

Content

- 2.2.1 Basic programming constructs/control structures
- 2.2.2 Use of subprograms/subroutines, including procedures and functions
- 2.2.3 Recursion

Learning outcomes

Candidates should be able to:

- (a) define and correctly use the following terms as they apply to procedural programming: statement, subroutine, procedure, function, parameter, loop
- (b) identify the three basic programming constructs used to control the flow of execution: sequence, selection and iteration
- (c) understand and use selection in pseudocode and a procedural programming language, including the use of IF statements and CASE/SELECT statements
- (d) understand and use iteration in pseudocode and a procedural programming language, including the use of count-controlled loops (FOR-ENDFOR loops) and condition-controlled loops (WHILE-ENDWHILE and REPEAT-UNTIL loops)
- (e) understand and use nested selection and nested iteration statements
- (f) understand, create and use subroutines (procedures and functions), including the passing of parameters and the appropriate use of the return value of functions
- (g) use subroutines to modularise the solution to a problem
- (h) identify and use recursion to solve problems; show an understanding of the structure of a recursive subroutine, including the necessity of a stopping condition
- (i) trace the execution of a recursive subroutine
- (j) discuss the relative merits of iterative and recursive solutions to the same problem

2.3 Data types and data structures

Content

- 2.3.1 Data types: integer, real, Boolean, character, string
- 2.3.2 Data structures: arrays (one- and two-dimensional), records
- 2.3.3 Storing, retrieving and searching for data in files (serial, sequential, random)

Learning outcomes

Candidates should, when writing a program in a procedural language, be able to:

- (a) define and use different data types, e.g. integer, real, Boolean, character and string
- (b) declare and use arrays (one- and two-dimensional) for solving simple problems (this should include initialising arrays, reading data into arrays and performing a simple serial search on a one-dimensional array)
- (c) design and implement a record format
- (d) estimate the size of a file from its structure and the number of records
- (e) store, retrieve and search for data in files
- (f) use the facilities of a procedural language to perform file operations (open, read, write, update, insert, append and close) on files and use appropriate file handling functions

2.4 Common facilities of procedural languages

Content

- 2.4.1 Assignment statements
- 2.4.2 Arithmetic, relational and Boolean operations
- 2.4.3 String manipulation
- 2.4.4 Input and output facilities

Learning outcomes

Using an appropriate procedural programming language, candidates should be able to:

- (a) understand and use assignment statements
- (b) understand arithmetic operators including operators for integer division (+, −, *, /, MOD and DIV) and use these to construct expressions
- (c) understand a range of relational operators, e.g. =, <, <=, >, >= and <> and use these to construct expressions
- (d) understand the Boolean operators AND, OR, and NOT and use these to construct expressions
- (e) understand the effects of the precedence of standard operators and the use of parentheses to alter the order of evaluation
- (f) evaluate expressions containing arithmetic, relational and Boolean operators and parentheses
- (g) understand and use a range of operators and built-in functions for string manipulation, including location (LOCATE), extraction (LEFT, MID, RIGHT), comparison, concatenation, determining the length of a string (LENGTH) and converting between characters and their ASCII code (ASCII and CHAR)
- (h) understand that relational operations on alphanumeric strings depend on binary codes of the characters
- (i) input and validate data
- (j) output data onto screen/file/printer, formatting the data for output as necessary

2.5 Writing maintainable programs

Content

- 2.5.1 Declaring and using variables and constants
- 2.5.2 Self-documented code, including identifiers, annotation and formatting

Learning outcomes

Using an appropriate procedural programming language, candidates should be able to:

- (a) define, understand and use the following terms correctly as they apply to programming: variable, constant, identifier, reserved word/keyword
- (b) declare variables and constants, understanding the effect of scope and issues concerning the choice of identifier (including the need to avoid reserved words/keywords)
- (c) select and use meaningful identifier names
- (d) initialise variables appropriately, before using them
- (e) annotate the code with comments so that the logic of the solution can be followed
- (f) use indentation and formatting to show clearly the control structures within the code

2.6 Testing and running a solution

Content

- 2.6.1 Types of programming errors
- 2.6.2 Testing strategies and test data
- 2.6.3 Debugging
- 2.6.4 Installation and execution

Learning outcomes

When developing software to solve a problem, candidates should be able to:

- (a) describe types of errors in programs (syntax, logic and run-time errors) and understand how and when these may be detected
- (b) describe testing strategies including white box testing, black box testing, alpha testing, beta testing and acceptance testing
- (c) select suitable test data for a given problem, including normal, borderline and invalid data
- (d) perform a dry run on a given algorithm, using a trace table
- (e) describe the use of a range of debugging tools and facilities available in program development environments including translator diagnostics, breakpoints, stepping, and variable check/watch

Section 3: System software mechanisms, machine architecture, database theory, programming paradigms and integrated information systems

The content includes:

- the functions of operating systems
- the functions and purposes of translators
- computer architectures and the fetch-execute cycle
- data representation, data structures and data manipulation
- programming paradigms
- databases
- simulation and real-time processing
- networking

Recommended prior knowledge

Candidates should have studied Section 1.

3.1 The functions of operating systems

Content

- 3.1.1 Features of operating systems
- 3.1.2 Scheduling (multi-programming)
- 3.1.3 Interrupt handling
- 3.1.4 Job queues and priorities
- 3.1.5 Memory management
- 3.1.6 Spooling
- 3.1.7 Modern personal computer operating systems

Learning outcomes

Candidates should be able to:

- (a) describe the main features of operating systems, including memory management and scheduling algorithms
- (b) explain how interrupts are used to obtain processor time and how processing of interrupted jobs may later be resumed (typical sources of interrupts should be identified and any algorithms and data structures should be described)
- (c) define and explain the purpose of scheduling, job queues, priorities and how they are used to manage job throughput
- (d) explain how memory is managed in a typical modern computer system (virtual memory, paging, partitions and segmentation)
- (e) describe spooling, explaining why it is used
- (f) describe the main components of a typical PC operating system, including the file allocation table (FAT) and boot file

3.2 The functions and purposes of translators

Content

- 3.2.1 Types of translators
- 3.2.2 Lexical analysis
- 3.2.3 Syntax analysis
- 3.2.4 Code generation
- 3.2.5 Linkers and loaders

Learning outcomes

Candidates should be able to:

- (a) understand the relationship between assembly language and machine code
- (b) describe how an assembler produces machine code from assembly language
- (c) describe the difference between interpretation and compilation
- (d) describe what happens during lexical analysis
- (e) describe what happens during syntax analysis
- (f) explain the code generation phase, and understand the need for optimisation
- (g) explain the purpose of linkers and loaders, and describe the use of library program routines
- (h) explain how errors are recognised and handled during compilation

3.3 Computer architectures and the fetch-execute cycle

Content

- 3.3.1 Von Neumann architecture
- 3.3.2 Registers: purpose and use
- 3.3.3 Fetch-execute cycle
- 3.3.4 Parallel processors

Learning outcomes

Candidates should be able to:

- (a) describe basic Von Neumann architecture, identifying the need for and the uses of special purpose registers in the functioning of a processor
- (b) describe in simple terms the fetch/execute cycle and the effects of the stages of the cycle on specific registers (Program Counter, Memory Address Register, Memory Data Register, Current Instruction Register, Index Register and Accumulator)
- (c) explain the need for and the use of buses to convey data (data, address and control buses)
- (d) discuss parallel processing systems (co-processor, parallel processor and array processor), their uses, their benefits and their drawbacks

3.4 Data representation, data structures and data manipulation

Content

- 3.4.1 Number systems
- 3.4.2 Floating point binary
- 3.4.3 Normalisation of floating point binary numbers
- 3.4.4 Implementation of data structures, including linked lists, stacks, queues and trees
- 3.4.5 Searching and sorting

Learning outcomes

Candidates should be able to:

- (a) express numbers in binary coded decimal (BCD) and hexadecimal
- (b) describe and use two's complement and sign-and-magnitude to represent positive and negative integers
- (c) perform integer binary addition
- (d) demonstrate an understanding of binary floating point representation of a real number
- (e) normalise the floating point representation of a number
- (f) discuss the trade-off between accuracy and range when representing numbers in floating point form
- (g) describe algorithms for the insertion, retrieval and deletion of data items stored in linked-list, binary tree, stack and queue structures
- (h) explain the difference between static and dynamic implementation of data structures, highlighting the benefits and drawbacks of each
- (i) explain the difference between a binary search and a sequential search, highlighting the advantages and disadvantages of each
- (j) use algorithms for implementing insertion sort and quick sort methods, and be able to explain the difference between them
- (k) describe the use of a binary tree to sort data
- (l) describe how data files are merged

3.5 Programming paradigms

Content

- 3.5.1 Types of languages and typical applications
- 3.5.2 Features of different types of language
- 3.5.3 Methods for defining syntax

Learning outcomes

Candidates should be able to:

- (a) describe the characteristics of a variety of programming paradigms (low-level, object-oriented, declarative and procedural)
- (b) explain, with examples, the terms object-oriented, declarative and procedural as applied to high-level languages
- (c) explain how functions, procedures and their related variables may be used to develop a program in a structured way, using stepwise refinement
- (d) describe the use of parameters, local and global variables as standard programming techniques
- (e) explain how a stack is used to handle procedure calling and parameter passing
- (f) discuss the concepts and, using examples, show an understanding of data encapsulation, classes and inheritance when referring to object-oriented languages
- (g) interpret and create class and object diagrams
- (h) discuss the concepts and interpret examples, showing an understanding of backtracking, instantiation and satisfying goals when referring to declarative languages
- (i) explain the concepts of immediate, direct, indirect, indexed and relative addressing of memory when referring to low-level languages
- (j) explain the need for, and be able to apply, BNF (Backus-Naur form) and syntax diagrams
- (k) explain the need for reverse Polish notation
- (l) convert between reverse Polish notation and the infix form of algebraic expressions using trees and stacks

Note: Candidates will **not** be expected to use any particular form to present algorithms, but should be able to write procedural algorithms in some form.
 Candidates will **not** be expected to write code in the examination.
 Candidates will be expected to understand and use a given algorithm in pseudocode.
 Candidates will be expected to understand and apply a given set of assembly language instructions.

3.6 Databases

Content

- 3.6.1 Database design
- 3.6.2 Normalisation and data modelling
- 3.6.3 Methods and tools for analysing and implementing database design
- 3.6.4 Control of access to relational database elements

Learning outcomes

Candidates should be able to:

- (a) describe flat files and relational databases
- (b) design a simple relational database to third normal form (3NF)
- (c) draw and interpret entity-relationship (E-R) diagrams
- (d) explain the benefits that using a relational database gives over flat files
- (e) define and explain the purpose of primary, secondary and foreign keys
- (f) explain the importance of varying the access allowed to database elements at different times and for different categories of user
- (g) describe the structure of a database management system (DBMS), including the function and purpose of the data dictionary, data description language (DDL) and data manipulation language (DML)

Note: Candidates will be expected to understand a simple DML script.

3.7 Simulation and real-time processing

Content

- 3.7.1 Applications of real-time computing (process control)
- 3.7.2 The feedback loop; input and output; sensors and actuators
- 3.7.3 The use of robots
- 3.7.4 Uses of simulation
- 3.7.5 Variation of parameters and conditions; time scales and intervals

Learning outcomes

Candidates should be able to:

- (a) describe real-time applications (process control)
- (b) explain the use of sensors for detecting physical signals (temperature, pressure, motion, light intensity)
- (c) explain the use of actuators
- (d) demonstrate an understanding of the use of robots in a variety of situations such as the manufacturing process or hazardous environments
- (e) explain the reasons for simulation, such as to change time-scales and/or save costs and/or avoid danger
- (f) discuss the advantages of simulation in testing the feasibility of a design

3.8 Networking

Content

- 3.8.1 Data transmission
- 3.8.2 Network components
- 3.8.3 Common network environments
- 3.8.4 Issues of security
- 3.8.5 Encryption and authentication techniques

Learning outcomes

Candidates should be able to:

- (a) demonstrate awareness of different media for transmitting data and their carrying capabilities
- (b) explain the different purposes of network components, including switches, routers, hubs, network interface cards (NIC) and modems
- (c) discuss common network environments, such as intranets, the Internet and other open networks, their facilities, structure and ability to exchange information using appropriate software and techniques
- (d) discuss the problem of maintaining security of data on an open network and practical techniques to address the issue
- (e) explain the need for encryption, authorisation and authentication techniques

Section 4: Computing project

The project is a substantial piece of work requiring analysis and design over an extended period of time, which is organised, evaluated and presented in a report.

Candidates choose, in conjunction with their teacher, a well-defined user-driven problem which enables them to demonstrate their skills in analysis, design and software development, including programming, testing, installation, documentation and evaluation. Problems should be selected that allow candidates to demonstrate their programming skills.

Projects should be chosen to demonstrate the integrative aspects of the work and should avoid needless repetition of the demonstration of a given skill. Each candidate must submit a report on their piece of work, supported by evidence of software development including programming and testing.

The teacher marks the projects using the marking criteria in the *Guidance on Marking the Computing Project* section of this syllabus, then moderation takes place following Cambridge procedures.

4.1 Report (3 marks)

A report presenting the Coursework as specified in 4.2 to 4.6.

Content

- 4.1.1 Organise the report into sections as given in the syllabus
- 4.1.2 Word process the report
- 4.1.3 Documentation of each stage of the development

Learning outcomes

Candidates should be able to:

- (a) organise the report
- (b) use word processing features where appropriate including checking of spelling and grammar
- (c) include the evidence specified in 4.2 to 4.6

4.2 Definition, investigation and analysis (11 marks)

Explanation of the problem to be solved, the user's requirements and how they were obtained. There should be a clear statement of requirements, agreed with the prospective client.

Content

- 4.2.1 Define a problem
- 4.2.2 Investigate the current system
- 4.2.3 Record findings
- 4.2.4 Analyse findings
- 4.2.5 Identify problems/inefficiencies with current system
- 4.2.6 Specify requirements: user, hardware, software

Learning outcomes

Candidates should be able to:

- (a) define the nature of the problem to be solved
- (b) use appropriate methods to investigate the problem and to gather information; these may include questionnaires, observation, meetings and document collection, but must include an interview with the client
- (c) record information/data and gather sample documents currently used
- (d) identify the current processes and current data structures
- (e) analyse the data and processes: candidates will be expected to use appropriate techniques such as structure diagrams/dataflow diagrams/system flowcharts to illustrate their analysis
- (f) specify inefficiencies and problems apparent from the information collection
- (g) derive the client's and information requirements of the system
- (h) specify the required hardware and give reasons for their choice
- (i) specify the required software and give reasons for their choice
- (j) develop and document a clear requirement specification

4.3 Design (12 marks)

Detailed system design including data structures, input-output format and processes involved, and testing required. There should be a clear design specification.

Content

- 4.3.1 Overview including an agreed set of objectives
- 4.3.2 Output design
- 4.3.3 Input design
- 4.3.4 Data structures/model
- 4.3.5 Process model
- 4.3.6 Test plan

Learning outcomes

Candidates should be able to:

- (a) agree a set of objectives with the client
- (b) design and document report layouts, screen displays and/or other forms of output, drawing up detailed models of the proposed interface
- (c) design and document data capture forms and/or screen layouts
- (d) design and document, using appropriate techniques (for example, normalisation/E-R models), the data structures necessary to solve the inefficiencies/problems indicated in the requirements specification
- (e) design and document an algorithm/pseudocode/top-down diagram or other form of process model which is/are necessary for the solution of the problem
- (f) design and document a test plan that includes test data and expected outcomes

4.4 Software development, programming, testing and installation (18 marks)

A software solution that includes some programming code using a stand-alone programming language or program embedded within application software (e.g. VBA used as the front end of a database solution written by the candidate). A comprehensive test plan is developed from the design, which should show that the system works to the satisfaction of the client by providing comprehensive functional testing, both alpha and beta, of the solution. The test plan should be clearly cross-referenced to the agreed set of objectives to provide evidence that the system has been tested during development and by the client.

Content

- 4.4.1 Software development
- 4.4.2 Programming
- 4.4.3 Testing a software solution
- 4.4.4 Planning for installation and use
- 4.4.5 Client and user testing

Learning outcomes

Candidates should be able to:

- (a) implement the proposed process model using a programming language and possibly the facilities of a software package
- (b) develop the data structures of the design using the appropriate features of a software package and programming language
- (c) develop inputs/outputs appropriate to the design of the solution
- (d) illustrate how the software solution evolves
- (e) test the software solution
- (f) produce detailed output from the testing, cross-referencing to the test plan
- (g) test the software solution with the client and user, providing documented evidence that the solution works, and devise a strategy for its installation

4.5 Documentation (10 marks)

The **System Maintenance Documentation** should include an explanation of the structure of the solution. All the necessary information about the system that would allow someone else to maintain and develop it should be included, for example, backup procedures/cycles, annotated code/modules, data structures used, and must include an element of adaptive maintenance in order to provide some future-proofing of the solution.

The **User Guide** should include step-by-step instructions for operating all aspects of the system, including a means of dealing with any errors that may occur. It should also include appropriate "Help" and messages within the software solution, and be present in the form of a hypertext document.

Content

- 4.5.1 System Maintenance Documentation
- 4.5.2 User Guide

Learning outcomes

Candidates should be able to:

- (a) develop system maintenance documentation
- (b) develop a detailed user guide

4.6 Evaluation (6 marks)

Discussion of the degree of success in meeting the original objectives as specified in the requirements specification, ease of the use of the package, acceptability to the client (including where possible a letter of acceptance from the client and reference to client and user testing results).

Content

4.6.1 Evaluate results against the agreed set of objectives

4.6.2 Evaluate the results of client and user testing

Learning outcomes

Candidates should be able to:

- (a) evaluate the final system against the criteria described in the agreed set of objectives
- (b) evaluate the client's and user's responses to testing the system

7. Coursework

Section 4: Computing project (60 marks)

This unit assesses candidates' ability to develop a computer-based solution to a real life problem requiring the skills of analysis, design, programming, development, testing, implementation and evaluation.

Candidates should formulate the task in negotiation with their teacher. If Centres are uncertain about the appropriateness of a problem they should seek advice from Cambridge.

Assessment and moderation

All coursework is marked by the teacher and internally standardised by the Centre. Coursework is then submitted to Cambridge by the specified date.

The internally moderated marks for all candidates must be recorded on the Coursework Assessment Summary Form. This form, and the instructions for completing it, may be downloaded from **www.cie.org.uk/samples**. The database will ask you for the syllabus code (i.e. 9691) and your Centre number, after which it will take you to the correct form. Follow the instructions when completing the form.

The purpose of moderation is to ensure that the standard for the award of marks in coursework is the same for each Centre, and that each teacher has applied the same standards appropriately across the range of candidates within the Centre.

Minimum coursework requirements

If a candidate submits no work for a coursework unit, then the candidate should be indicated as being absent from that unit on the coursework mark sheets submitted to Cambridge.

If a candidate completes any work for the coursework unit, then the work should be assessed according to the criteria and marking instructions, and the appropriate mark awarded (which may be zero).

Authentication

As with all coursework, **the teacher must be able to verify that the work submitted for assessment is the candidate's own work. Sufficient work must be carried out under direct supervision to allow the teacher to authenticate the coursework marks with confidence.**

Cambridge is happy to rely on the professionalism of teachers to ensure fairness with this work.

Differentiation

In the question papers, differentiation is achieved by setting questions which are designed to assess candidates at their appropriate levels of ability, and which are intended to allow all candidates to demonstrate what they know, understand and can do.

In coursework, candidates should choose their project problem so that the work enables them to display positive achievement and to demonstrate their full range of abilities.

Recording candidates' marks

Candidates' marks must be recorded on the Individual Candidate Record Card produced by Cambridge. These forms, and the instructions for completing them, may be downloaded from **www.cie.org.uk/samples**. The database will ask you for the syllabus code (i.e. 9691) and your Centre number, after which it will take you to the correct forms. Follow the instructions when completing each form.

Please copy the Marking Grid at the back of this syllabus document and submit with the Computing Project. The Marking Grid for the Computing Project should be attached to each candidate's submission.

8. Appendix

8.1 Guidance on selecting the Computing Project

The selection of the problem for which a computerised system is to be designed and implemented is extremely important. It should be chosen by the candidate in consultation with the teacher, and should always involve a client, who requires the solution to the problem, and user(s), the person who is going to use the computerised system. The client and the user may be the same person, e.g. if a sole-trader's business requires a computerised system.

It is important to stress that the candidate should endeavour to produce a system which will solve a given problem sensibly, within the constraints of resources available to the candidate.

Since the computing project seeks to assess the systems analysis section of the specification in a practical manner, candidates should not produce a system from their own limited knowledge of the requirements of the system. The client has to be someone who is willing to be involved in the project:

- in the analysis of the problem, where the client's requirements are obtained; this may take the form of recorded interviews with the candidate
- at the software development, testing and implementation stages, where the client and/or user is involved in 'prototyping'
- at the evaluation stage, where the client is involved in checking that the system is completed as specified and, leading on from this, is then willing to write a letter of acceptance of the system, including any criticisms of it.

In this way, candidates can be encouraged to look beyond school or college life into the businesses and companies in the community of the surrounding area. The emphasis is on analysing an existing system, and producing a computer-based solution to fit the needs of a client.

At the end of the project, candidates should submit a concisely written and well laid out report, which should be word processed.

The solution must be implemented using a programming language and any of the following that are appropriate: pre-written modules or toolkits, applications software and programmable packages. Very brief descriptions of any programming languages or software packages used, together with reasons for their selection, should be included in the report.

For the programming the candidate should:

- annotate listings
- explain each section of the program with appropriate algorithm descriptions, which should be language independent
- define variables by name, type and function where appropriate
- define clearly and identify the purpose of subroutines and procedures.

Where part of the solution has been produced with a software package that has not involved programming, the candidate should:

- explain each section of the solution with appropriate algorithm descriptions
- define the purpose and inter-relationship of modules within the system
- clearly annotate the results produced.

The projects should be documented in a report that contains the title, a contents list, and is set out in the sub-sections identified in the 'Guidance on Marking the Computing Project'.

Appropriate evidence of development, testing and demonstration of a working system, such as screen dumps or photographs of screen layouts and printouts, paper based user documentation and a letter from the client to say that the system has been developed satisfactorily, must be included in the report.

Candidates should not submit magnetic, optical or solid state media as supporting evidence.

The computing project must involve programming and may involve the tailoring of generic software packages and may also involve the choosing and installing of hardware. It is not intended that any method of solution is better than another merely that the solution must be one that suits the problem that is being solved.

8.2 Guidance on marking the Computing Project

Computing Projects are assessed as follows:

| | |
|---|-----------------|
| Quality of report | 3 marks |
| Definition, investigation and analysis | 11 marks |
| Design | 12 marks |
| Software development, programming, testing and installation | 18 marks |
| Documentation | 10 marks |
| Evaluation | 6 marks |
| Total | 60 marks |

(a) Quality of report [Total 3 marks]

A candidate should produce a well ordered report that covers all the information from the sections set out below.

| | |
|--|---------|
| Evidence for most sections is included; there may be errors of spelling, punctuation and grammar. | 1 mark |
| Evidence for all sections is included, the report is well ordered and there are few errors of spelling, punctuation and grammar. | 2 marks |
| The report is complete, well organised with good use of illustrations, and there may be a few minor errors of spelling, punctuation and grammar. | 3 marks |

(b) Definition, investigation and analysis [Total: 11 marks]

(i) Definition – nature of the problem [3 marks]

A candidate should not expect the examiners to be familiar with the theory and practice in the area of the chosen system. There should be a brief description of the organisation (for example, firm or business) involved and the current methods used in the chosen areas that may form the basis of the project. A clear statement of the origins and form of data should be given. At this stage the exact scope of the project may not be known and it may lead to the arranging of an interview with the client.

| | |
|---|---------|
| Description of the organisation. | 1 mark |
| Description of the organisation and the methods currently used in the area of the chosen project. | 2 marks |
| Full description of the organisation and methods currently in use in the area of the chosen project, with a description of the origin of the data to be used and some indication of the form that data takes. | 3 marks |

(ii) Investigation and analysis [8 marks]

This section is the 'systems analysis'. The candidate should describe how the client requirements were ascertained (possibly by long discussions with the users: question and answer sessions should be recorded and outcomes agreed). A clear requirements specification should be defined. Alternative outline solutions should be discussed and evaluated against one another.

| | |
|--|-----------|
| Some elements have been discussed but little or no client involvement. | 1–2 marks |
| Some evidence that an attempt has been made to interview the client and some recording of it has been made. An attempt has been made to develop a requirement specification based on the information collected. | 3–4 marks |
| Good client involvement and recording of the interview(s). Most of the necessary items have been covered including a detailed discussion of alternative approaches. A requirements specification based on the information collected is present but with some omissions. | 5–6 marks |
| Excellent client and user involvement with detailed recording of the client's requirements. Alternative approaches have been discussed in depth. The report demonstrates a thorough analysis of the system to be computerised. A detailed requirements specification based on the information collected has been produced. | 7–8 marks |

(c) Design [Total: 12 marks]

(i) Nature of the solution [8 marks]

A detailed systems design (including diagrams as appropriate) should be produced and agreed with the client. Proposed record, file and data structures should be described and design limitations should be included. Design of data capture forms, input formats (with examples of screen layouts) and output formats should be included here where relevant. Process designs and a test plan for the system should also be included. The test plan should contain test data and the expected results for that data. An agreed set of objectives should also be included. These items are the design specifications, which should be agreed with the client.

| | |
|---|-----------|
| Some vague discussion of what the system will do with a brief diagrammatic representation of the new system. | 1–2 marks |
| The major objectives of the new system have been adequately summarised, <i>but omissions have been made</i> . There is a brief outline of a design specification, including mock-ups of inputs and outputs, process model described (including a diagram: structure diagram, dataflow diagram or system flowchart). However, there is a lack of completeness with omissions from the process model, inputs and outputs. Data structures have been identified but there may be inadequate detail. The test plan may be incomplete. | 3–4 marks |
| A clear set of objectives has been defined and a full design specification is included, but there may be some errors or logical inconsistencies, for example validation specified may be inadequate or field lengths incorrect. There is clear evidence that a response to the design has been obtained from the client, and any comments have been acted upon. | 5–6 marks |
| A clear set of agreed objectives with a detailed and complete design specification, which is logically correct. There are also detailed written descriptions of any processes/modules and a clear, complete definition of any data structures. The specification is sufficient for someone to pick up, develop and test an end result using the software and hardware specified in the requirements specification. | 7–8 marks |

(ii) Intended benefits [2 marks]

There should be some discussion of the relative merits of the intended system and of the previous mode of operation. This may include any degree of generality beyond the original scope of the system.

| | |
|--|---------|
| One valid benefit of the new system has been identified and explained. | 1 mark |
| The benefits of the new system have been comprehensively described. | 2 marks |

(iii) Limits of the scope of the solution [2 marks]

This may include volume (sizing limitations) and limitations of the facilities used. For full marks there must be some estimate of the size of the files required for the implemented system.

| | |
|---|---------|
| A discussion of what the system limitations are. | 1 mark |
| A detailed description of the system limitations has been given, including the estimate of the size of the files required for the implemented system. | 2 marks |

(d) Software development, programming, testing and installation [Total: 18 marks]

(i) Development [4 marks]

A technical description of how the solution relates to the design specification produced and agreed with the user should be included.

| | |
|--|-----------|
| Program listings or evidence of tailoring of a software package is provided in the form of printouts. The developed solution does not fulfil the design specification. A teacher may award 1 mark if they have been shown the system working satisfactorily and there is no hard evidence in the project report. | 1 mark |
| Program listings or evidence of tailored software packages are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate, detailing their purpose. There is some annotation evident to illustrate how the package was tailored for a particular purpose or to indicate the purpose of sections of code in a program listing. The developed solution has logical flaws and does not fulfil the design specification. | 2–3 marks |
| Program listings or evidence of tailored software packages are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate, detailing their purpose. There is a full set of printouts showing input and output as well as data structures. The developed solution does fulfil the design specification. | 4 marks |

(ii) Programming [5 marks]

There should be clearly set out program listings that demonstrate the technical competence of the candidate. Candidates should make good use of the facilities of a procedural programming language as part of their solution.

| | |
|--|-----------|
| A program listing showing code written by the candidate is included. | 1–2 marks |
| Some technical competence in programming shown by a program listing that makes use of meaningful identifier names, indentation and formatting to show the control structures used. The code should be annotated with some comments so that the logic of the solution can be followed. | 3–4 marks |
| Good technical competence in programming shown by a self-documented program listing that makes good use of meaningful identifier names, indentation and formatting to show the control structures used. The code should be annotated with comments so that the logic of the solution can be easily followed. | 5 marks |

(iii) Testing [5 marks]

An attempt should be made to show that all parts of the system have been tested, including those sections dealing with unexpected or invalid data as well as borderline cases. Showing that many other cases of test data are likely to work – by including the outputs that they produce – is another important feature. Evidence of testing is essential. Comments by teachers and others are of value, but the test plan must be supported by evidence in the report of a properly designed testing process. The examiner must be left in no doubt the system actually works to the satisfaction of the client. This evidence may be in the form of hardcopy output and screen dumps.

| | |
|---|-----------|
| A collection of hardcopy test run outputs with no test plan, or a test plan with no hardcopy evidence may also be present. A teacher may award 1 mark if they have been shown the system working satisfactorily and there is no hard evidence in the project report. | 1 mark |
| There is little evidence of testing with a badly developed test plan with clear omissions. There is no description of the relationship between the structure of the development work and the testing in evidence. | 2 marks |
| There should be hardcopy evidence from at least eight different test runs cross-referenced to the test plan. However, not all cases have been tested. | 3–4 marks |
| Evidence of each test run cross-referenced to the test plan is present in the report. Testing should include as many different paths through the system as is feasible, including valid, invalid and borderline cases. Marks may be lost for lack of evidence of a particular test run. | 5 marks |

(iv) Installation [4 marks]

It is recognised that the client may not fully install and use the system, although this is the ultimate aim. However, to score any marks in this section, there must be some evidence that the client has seen the system in operation. This can be done in a number of ways: such as by inviting the client to see the product and allowing the candidate to demonstrate the system, or by taking the system to the client involved. There should be an installation plan written, including details of system changeover, training required and details of user testing.

| | |
|--|-----------|
| Details of system changeover have been documented. Some evidence of client and/or user testing is given, usually by questionnaire or written comments by fellow students or others who were not directly involved in the development of the system. | 1 mark |
| An implementation plan with details of system changeover and training required. There is written evidence available from the client indicating that they have seen the system in operation. | 2–3 marks |
| A clear and detailed implementation plan, including planned system changeover, training required and detailed stages of user testing. There is written evidence available from the client and/or user that they have tested the system and agree with the strategy for implementation. | 4 marks |

(e) Documentation [Total: 10 marks]

(i) System maintenance documentation [4 marks]

Much of the documentation will have been produced as a by-product of design and development work and also as part of writing up the report to date. The contents of the manual should, where relevant, include the following: record, file and data structures used; data dictionary; data flow (or navigation paths); annotated program listings; detailed flowcharts; details of the algorithms used and adaptive maintenance to provide for some future-proofing.

All parts of the manual should be fully annotated, since this is important for subsequent maintenance of the system. The specifications of the hardware and software on which the system can be implemented should be included.

| | |
|--|-----------|
| Some items are present with some annotation attempted. | 1–2 marks |
| One or two omissions, but the rest is present and annotation is used sensibly. | 3–4 marks |

(ii) User guide [6 marks]

Clear guidance, as friendly as possible, should be given to the user for all operations that they would be required to perform. These would include input format with screen displays, print options, backups (file integrity routines), security of access to data and a guide to common errors that may occur. (Note: the candidate would **not** be required to copy out large volumes of any underlying software's user guide, but to produce a non-technical and easy-to-follow guide for someone with little computer knowledge.) Some mention here of the relationship between items of software and the data they deal with may be relevant.

The user guide should be well presented with an index and, where necessary, a glossary of the terms used. Alternatively, an electronic guide could be based around hypertext links (screen dumps will be required).

| | |
|--|-----------|
| An incomplete guide, perhaps with no screen displays. Some options briefly described, but difficult for the user to follow. | 1–2 marks |
| All but one or two options fully described (for example, backup routines not mentioned). Mostly, the options are easy for the user to follow with screen displays. | 3–4 marks |
| A full user guide with all options described well presented (possibly as booklet), with an index and a glossary. No omission of any of the options available (including backup routines, guide to common errors). Marks may be lost for inadequate descriptions of some options. For full marks, good on-screen help should exist where this is a sensible option, and be present in the form of a hypertext document. | 5–6 marks |

(f) Evaluation [Total: 6 marks]

(i) Discussion of the degree of success in meeting the original objectives [3 marks]

This discussion should demonstrate the candidate's ability to evaluate the effectiveness of the completed system. The agreed set of objectives should be matched to the achievements, taking into account the limitations. Client and user evaluation is also essential, and should arise from a questionnaire or, preferably, direct evaluation. For full marks it is important that the user provides sets of data as they are likely to occur in practice, and that the results arising from such data be taken into account. This data is typical data rather than test data, and it may show up faults or problems that the candidate's own test data failed to find.

| | |
|---|---------|
| Some discussion about the success, or otherwise, of the work, but with no reference to the specification set out in (c)(i) . | 1 mark |
| Some discussion about a number of the objectives set out in (c)(i) , but some omissions or inadequate explanation of success or failure. | 2 marks |
| A full discussion, taking each objective mentioned in (c)(i) and explaining the degree of success in meeting them, indicating where in the project evidence can be found to support this, or giving reasons why they were not met. | 3 marks |

(ii) Evaluation of the client's and user's response to the system [3 marks]

It is important that neither the client nor the user is assumed to be an expert in computer jargon, so some effort must be made to ensure that the system is user-friendly. It will be assumed that the client will have considerable knowledge of the underlying theory of the business being computerised. Clarity of menus, clear on-screen help and easy methods of inputting data are all examples of how the system can be made user-friendly. Here, marks are awarded for the degree of satisfaction that the client indicates in the acceptance procedure. Could the system or its results be used? Was the system specification achieved? Do any system faults still exist? The candidate should evaluate the client's response to the final version of the system.

It is important that the client and the user become active participants in this section, and that their responses are reported and evaluated by the candidate.

| | |
|---|---------|
| Some effort has been made to make the system user-friendly, but the user still has difficulty using the system. | 1 mark |
| The system is mostly user-friendly, but there is room for improvement (e.g. no on-screen help has been provided). The client indicates that the system could be used, but there are some faults which need to be rectified. | 2 marks |
| A fully user-friendly system has been produced. The client indicates that the system fully meets the specification given in section (b) , and there are no known faults in the system. | 3 marks |

Paper 4: Marking Grid (attach to Project)

Computing Project marking details

| | | | |
|------------------------|--|---------------------------|----------------------|
| | | Total Project mark | [__/60 marks] |
| Candidate name: | | Candidate number: | |
| Centre name: | | Centre number: | |

| | |
|------------------------------|---------------------|
| (a) Quality of report | [__/3 marks] |
|------------------------------|---------------------|

| | | Mark | Comments |
|---|--|-------------|-----------------|
| 1 | Evidence for most sections is included; there may be errors of spelling, punctuation and grammar. | | |
| 2 | Evidence for all sections is included, the report is well ordered, and there are few errors of spelling, punctuation and grammar. | | |
| 3 | The report is complete, well organised with good use of illustrations, and there may be a few minor errors of spelling, punctuation and grammar. | | |

| | |
|---|----------------------|
| (b) Definition, investigation and analysis | [__/11 marks] |
|---|----------------------|

(i) Definition – nature of the problem **[__/3 marks]**

| | | Mark | Comments |
|---|---|-------------|-----------------|
| 1 | Description of the organisation. | | |
| 2 | Description of the organisation and the methods currently used in the area of the chosen project. | | |
| 3 | Full description of the organisation and methods currently in use in the area of the chosen project, with a description of the origin of the data to be used and some indication of the form that data takes. | | |

(ii) Investigation and analysis **[__/8 marks]**

| | | Mark | Comments |
|-----|---|-------------|-----------------|
| 1–2 | Some elements have been discussed but little or no user involvement. | | |
| 3–4 | Some evidence that an attempt has been made to interview the client and some recording of it has been made. An attempt has been made to develop a requirement specification based on the information collected. | | |
| 5–6 | Good client involvement and recording of the interview(s). Most of the necessary items have been covered including a detailed discussion of alternative approaches. A requirements specification based on the information collected is present but with some omissions. | | |
| 7–8 | Excellent client involvement with detailed recording of the client's requirements. Alternative approaches have been discussed in depth. The report demonstrates a thorough analysis of the system to be computerised. A detailed requirements specification based on the information collected has been produced. | | |

(c) Design

[___/12 marks]

(i) Nature of the solution

[___/8 marks]

| | | Mark | Comments |
|-----|---|------|----------|
| 1–2 | Some vague discussion of what the system will do with a brief diagrammatic representation of the new system. | | |
| 3–4 | The major objectives of the new system have been adequately summarised, but omissions have been made. There is a brief outline of a design specification, including mock-ups of inputs and outputs, process model described (including a diagram: structure diagram, dataflow diagram or system flowchart). However, there is a lack of completeness with omissions from the process model, inputs and outputs. Data structures have been identified but there may be inadequate detail. The test plan may be incomplete. | | |
| 5–6 | A clear set of objectives have been defined and a full design specification is included but there may be some errors or logical inconsistencies, for example validation specified may be inadequate or field lengths incorrect. There is clear evidence that a response to the design has been obtained from the client, and any comments have been acted upon. | | |
| 7–8 | A clear set of agreed objectives with a detailed and complete design specification, which is logically correct. There are also detailed written descriptions of any processes/modules and a clear, complete definition of any data structures. The specification is sufficient for someone to pick up, develop and test an end result using the software and hardware specified in the requirements specification. | | |

(ii) Intended benefits

[___/2 marks]

| | | Mark | Comments |
|---|--|------|----------|
| 1 | One valid benefit of the new system has been identified and explained. | | |
| 2 | The benefits of the new system have been comprehensively described. | | |

(iii) Limits of the scope of the solution

[___/2 marks]

| | | Mark | Comments |
|---|--|------|----------|
| 1 | A discussion of what the system limitations are. | | |
| 2 | A detailed description of the system limitations has been given, including the estimate of the size of the files required for the implemented system | | |

(d) Software development, programming, testing and installation

[___/18 marks]

(i) Development

[___/4 marks]

| | | Mark | Comments |
|-----|--|------|----------|
| 1 | Program listings or evidence of tailoring of a software package is provided in the form of printouts. The developed solution does not fulfil the design specification. A teacher may award 1 mark if they have been shown the system working satisfactorily and there is no hard evidence in the project report. | | |
| 2–3 | Program listings or evidence of tailored software packages are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate, detailing their purpose. There is some annotation evident to illustrate how the package was tailored for a particular purpose or to indicate the purpose of sections of code in a program listing. The developed solution has logical flaws and does not fulfil the design specification. | | |
| 4 | Program listings or evidence of tailored software packages are provided in the form of printouts. Data structures are illustrated as part of the listings where appropriate, detailing their purpose. There is a full set of printouts showing input and output as well as data structures. The developed solution does fulfil the design specification. | | |

(ii) Programming

[___/5 marks]

| | | Mark | Comments |
|-----|--|------|----------|
| 1–2 | A program listing showing code written by the candidate is included. | | |
| 3–4 | Some technical competence in programming shown by a program listing that makes use of meaningful identifier names, indentation and formatting to show the control structures used. The code should be annotated with some comments so that the logic of the solution can be followed. | | |
| 5 | Good technical competence in programming shown by a self-documented program listing that makes good use of meaningful identifier names, indentation and formatting to show the control structures used. The code should be annotated with comments so that the logic of the solution can be easily followed. | | |

(iii) Testing

[__/5 marks]

| | | Mark | Comments |
|-----|---|------|----------|
| 1 | A collection of hardcopy test run outputs with no test plan, or a test plan with no hardcopy evidence may also be present. A teacher may award 1 mark if they have been shown the system working satisfactorily and there is no hard evidence in the project report. | | |
| 2 | There is little evidence of testing with a badly developed test plan with clear omissions. There is no description of the relationship between the structure of the development work and the testing in evidence. | | |
| 3–4 | There should be hardcopy evidence from at least eight different test runs cross-referenced to the test plan. However, not all cases have been tested. | | |
| 5 | Evidence of each test run cross-referenced to the test plan is present in the report. Testing should include as many different paths through the system as is feasible, including valid, invalid and borderline cases. Marks may be lost for lack of evidence of a particular test run. | | |

(iv) Installation

[__/4 marks]

| | | Mark | Comments |
|-----|---|------|----------|
| 1 | Details of system changeover have been documented. Some evidence of client and/or user testing is given, usually by questionnaire or written comments by fellow students or others who were not directly involved in the development of the system. | | |
| 2–3 | An implementation plan with details of system changeover and training required. There is written evidence available from the client indicating that they have seen the system in operation. | | |
| 4 | A clear and detailed implementation plan including planned system changeover, training required and detailed stages of user testing. There is written evidence available from the client and/or user that they have tested the system and agree with the strategy for implementation. | | |

(e) Documentation

[__/10 marks]

(i) Systems maintenance documentation

[__/4 marks]

| | | Mark | Comments |
|-----|--|------|----------|
| 1–2 | Some items are present with some annotation attempted. | | |
| 3–4 | One or two omissions, but the rest is present and annotation is used sensibly. | | |

(ii) User Guide

[__/6 marks]

| | | Mark | Comments |
|-----|---|------|----------|
| 1–2 | An incomplete guide, perhaps with no screen displays. Some options briefly described but difficult for the user to follow. | | |
| 3–4 | All but one or two options fully described (for example, backup routines not mentioned). Mostly, the options are easy for the user to follow with screen displays. | | |
| 5–6 | A full user guide with all options described well presented (possibly as booklet) with an index and a glossary. No omission of any of the options available (including backup routines, guide to common errors). Marks may be lost for inadequate descriptions of some options. For full marks, good on-screen help should exist where this is a sensible option, and be present in the form of a hypertext document. | | |

(f) Evaluation

[__/6 marks]

(i) Discussion of the degree of success in meeting the original objectives

[__/3 marks]

| | | Mark | Comments |
|---|--|------|----------|
| 1 | Some discussion about the success, or otherwise, of the work, but with no reference to the specification set out in (c)(i). | | |
| 2 | Some discussion about a number of the objectives set out in (c)(i), but some omissions or inadequate explanation of success or failure. | | |
| 3 | A full discussion, taking each objective mentioned in (c)(i) and explaining the degree of success in meeting them, indicating where in the project evidence can be found to support this, or giving reasons why they were not met. | | |

(ii) Evaluation of the client's and user's response to the system

[__/3 marks]

| | | Mark | Comments |
|---|--|------|----------|
| 1 | Some effort has been made to make the system user-friendly, but the user still has difficulty using the system. | | |
| 2 | The system is mostly user-friendly, but there is room for improvement (e.g., no on-screen help has been provided). The client indicates that the system could be used, but there are some faults which need to be rectified. | | |
| 3 | A fully user-friendly system has been produced. The user indicates that the system fully meets the specification given in section (b), and there are no known faults in the system. | | |

9. Other information

Equality and inclusion

Cambridge International Examinations has taken great care in the preparation of this syllabus and assessment materials to avoid bias of any kind. To comply with the UK Equality Act (2010), Cambridge has designed this qualification with the aim of avoiding direct and indirect discrimination.

The standard assessment arrangements may present unnecessary barriers for candidates with disabilities or learning difficulties. Arrangements can be put in place for these candidates to enable them to access the assessments and receive recognition of their attainment. Access arrangements will not be agreed if they give candidates an unfair advantage over others or if they compromise the standards being assessed.

Candidates who are unable to access the assessment of any component may be eligible to receive an award based on the parts of the assessment they have taken.

Information on access arrangements is found in the *Cambridge Handbook* which can be downloaded from the website www.cie.org.uk/examsOfficers

Language

This syllabus and the associated assessment materials are available in English only.

Grading and reporting

Cambridge International A Level results are shown by one of the grades A*, A, B, C, D or E, indicating the standard achieved, A* being the highest and E the lowest. 'Ungraded' indicates that the candidate's performance fell short of the standard required for grade E. 'Ungraded' will be reported on the statement of results but not on the certificate. The letters Q (result pending), X (no results) and Y (to be issued) may also appear on the statement of results but not on the certificate.

Cambridge International AS Level results are shown by one of the grades a, b, c, d or e, indicating the standard achieved, 'a' being the highest and 'e' the lowest. 'Ungraded' indicates that the candidate's performance fell short of the standard required for grade 'e'. 'Ungraded' will be reported on the statement of results but not on the certificate. The letters Q (result pending), X (no results) and Y (to be issued) may also appear on the statement of results but not on the certificate.

If a candidate takes a Cambridge International A Level and fails to achieve grade E or higher, a Cambridge International AS Level grade will be awarded if both of the following apply:

- the components taken for the Cambridge International A Level by the candidate in that series included all the components making up a Cambridge International AS Level
- the candidate's performance on these components was sufficient to merit the award of a Cambridge International AS Level grade.

For languages other than English, Cambridge also reports separate speaking endorsement grades (Distinction, Merit and Pass), for candidates who satisfy the conditions stated in the syllabus.

Entry codes

To maintain the security of our examinations, we produce question papers for different areas of the world, known as 'administrative zones'. Where the component entry code has two digits, the first digit is the component number given in the syllabus. The second digit is the location code, specific to an administrative zone. Information about entry codes for your administrative zone can be found in the *Cambridge Guide to Making Entries*.

Cambridge International Examinations
1 Hills Road, Cambridge, CB1 2EU, United Kingdom
Tel: +44 (0)1223 553554 Fax: +44 (0)1223 553558
Email: info@cie.org.uk www.cie.org.uk

® IGCSE is the registered trademark of Cambridge International Examinations

© Cambridge International Examinations 2014

