Regulated

OCR Entry Level Certificate in Computer Science (R354)

Teachers' Handbook

Version 1: First assessment 2017

Contents

1   Introduction

1.1   What is the purpose of this handbook?

Our Entry Level Certificate in Computer Science has been redesigned for first teaching in September 2016. It sits with the new GCSE (9–1) in Computer Science, enabling us to offer a coherent package for you and your learners.

This is an exciting, broad-based specification for you and your learners, which aims to make Computer Science engaging for learners and encourage them to achieve more. The practical possibilities are wide ranging and should enable every learner to find things they can engage with.

This Teachers' Handbook plays a dual role with the specification itself. The 'Entry Level Certificate in Computer Science' specification and this Teachers' Handbook contain information on which the assessment is based; the specification indicates what content, skills and programming techniques need to be covered and is supported by the Teachers' Handbook. At all times therefore, the Teachers' Handbook should be read in conjunction with the specification. If there is any contradiction found between the Handbook and the specification it is the specification that takes precedence.

1.2 Overview of OCR Entry Level Certificate in Computer Science (R354)

Learners must complete all three parts of the subject content which makes up 100% of the assessment.

Content Overview	Assessment Overview	
Computer Systems **CS1** • Computer hardware • Computer software **CS2** • Computer memory and storage • Moral, legal, cultural and environmental concerns	Computer Systems 40 Marks 2x30 minute exam Total 1 hour Set by OCR Internally-assessed/ externally-moderated	**40%** of total Entry Level Certificate
Computational thinking, algorithms and programming **CTAP1** • Computational logic • Algorithms **CTAP2** • Programming techniques • Data representation	Computational Thinking, Algorithms and Programming 40 Marks 2x30 minute exam Total 1 hour Set by OCR Internally-assessed/ externally-moderated	**40%** of total Entry Level Certificate
Programming project • Planning a solution • Developing a solution • Testing a solution • Evaluating the succes of the solution	Programming Project 20 Marks Set by OCR Internally-assessed/ externally-moderated	**20%** of total Entry Level Certificate

1.3 How do I find out more information?

If you are already using OCR specifications you can contact us at: www.ocr.org.uk

If you are not already a registered OCR centre then you can find out more information on the benefits of becoming one at: www.ocr.org.uk

If you are not yet an approved centre and would like to become one, go to: www.ocr.org.uk

Want to find out more?

Ask subject specialists:

Email: computerscience@ocr.org.uk

Teacher support:

www.ocr.org.uk/computerscience

Phone: 01223 553998

Twitter: @OCR_ict

2 Curriculum guidance

The Entry Level Certificate in Computer Science should introduce learners to the fundamental concepts of computer science. Learners choose a programming project task from a range of OCR released programming projects and complete two written tests for other sections of the subject content. Please see the specification for detailed subject content.

It is important that candidates have sufficient depth of knowledge and understanding, combined with developed computational skills, to meet the demands of the internal assessment. The Entry Level Computer Science specification provides learners with opportunities to become more familiar with practical applications of computer science.

This course is designed to provide learners with a broad understanding of the world of Computer Science and encourage them to develop computational thinking, algorithm and programming skills. The scheme of assessment for the Entry Level Certificate in Computer Science will prepare learners for GCSE by covering some content that is similar to the GCSE (9–1) Computer Science

and this provides a firm foundation from which learners may make informed decisions about progressing on to a GCSE (9–1) Computer Science or alternatively, a Vocational IT qualification. This means that learners taking an Entry Level Certificate and learners taking the GCSE can be taught together – essentially meaning co-teachability.

The Entry Level Certificate may also be suitable for candidates who need to transfer mid-course from the GCSE (9–1) Computer Science qualification at any stage, so as to focus more on the basic knowledge of Computer Science.

As well as main-stream centres, the Entry Level Certificate will appeal to non-school centres such as young offender institutions, hospital school or Pupil Referral Units etc.

It is essential that guidance is followed by all centres to ensure the authenticity of work submitted and to guard against plagiarism. All practical work marked as part of the internal assessment should be undertaken in supervised conditions and marked with reference to the assessment criteria

2.1 Subject specific guidance

The subject content for R354 is made up of three sections that learners must complete in order to obtain an Entry Level Certificate in Computer Science. The examination of these three sections will be internally assessed and externally moderated.
There are four written tests in total, 30 minutes each, as listed below:

- Computer Systems – CS1 and CS 2 **OR** CS 3 and CS 4
- Computational Thinking, Algorithms and Programming – CTAP1 and CTAP2 **OR** CTAP3 and CTAP 4

It is important to note that CS1 and CS2 form one assessment of Computer Systems, and CS3 and CS4 form an alternative (re-sit) assessment of Computer Systems. Therefore, you must submit these papers as a unit and cannot mix. For example, you cannot submit CS1 and CS4 for Computer Systems. However, you may submit CS1 & CS2 for Computer Systems, and then CS3 & CS4 for Computational Thinking, Algorithms and Programming.

Submission of tests	Submit either			
Computer Systems	CS1 CS2	**or**	CS3 CS4	
	and either			
Computational thinking, algorithms and programming	CTAP1 CTAP2	**or**	CTAP3 CTAP4	

There are various methods for completing the assessment required in this qualification. Teachers can set the assessment tests for topics when they think the learners are ready. If the learners complete the assessment early, they can stop and the assessment can be marked.

In preparation for the assessment, teachers must allow sufficient teaching time to increase the learners' depth of knowledge and understanding of the subject content. The method of delivery is up to the teachers' discretion. The theory can be taught as individual sections before learners take the tests. Alternatively, it can be taught as a larger section where learners complete both CS1 and CS2 for a complete section for the computer systems or CTAP1 and CTAP2 for a complete section for the Computational Thinking, Algorithms and Programming.

The tests are available for downloading from Interchange. Tests are marked by the teacher, internally, using mark schemes provided by OCR and submitted to OCR, together with all other sections, as required in the specification. OCR will then externally verify the marks awarded. This is different from externally assessed tests.

Learners must complete a programming project, using any suitable programming language, from the 3 OCR released programming projects and submit this for internal moderation / marking. Learners should be taught all of the programming techniques before undertaking the programming project. The programming language used by learners may be:
- Drag and drop, for example:
    - Scratch
    - Gamemaker
    - Greenfoot
    - BYOB
    - AppInventor
- Textually derived, for example:
    - Python
    - Small Basic
    - Visual Basic /.NET
    - Java Script
    - Java
    - C# or other C family language
    - BASH
    - Delphi / Pascal / Lazarus

## 2.1.1   Guidance on the sections of the subject content:

The content of **Computer Systems** is broken down into four sub-sections of which two sub-sections are included in CS1 and the other two sub sections are included in CS2. There are two sets of test papers covering the 'Computer Systems' content. One set would be used as the test and the second set can be used as a re-sit, if required.
- CS1 and CS3 will cover the subject content from the Computer Hardware and Computer Software sub-sections
- CS2 and CS4 will cover the subject content from the Computer memory and storage and Moral, legal and environmental concerns sub-sections.

The content of the **Computational thinking, Algorithms and Programming** is broken down to four sub sections, of which two sub sections are included in CTAP1 and the other two sub sections are included in CTAP2. There are two sets of test papers covering the 'Computational thinking, Algorithms and Programming' content. One set would be used as the test and the second set can be used as a re-sit, if required.

- CTAP1 and CTAP3 will cover the subject content from the Computational Logic and Algorithms sub-sections
- CTAP2 and CTAP4 will cover the subject content from the Programming Techniques and Data Representation sub-sections.

The **Programming Project,** provided by OCR, will be released through Interchange for teachers to download. Learners may use a programming project from OCR set projects, or you may set one project from OCR released projects for a whole group of learners. An iterative process for developing a coded programming project will be most suitable for learners. The tasks will provide opportunities for the learners to demonstrate their practical ability in the skills outlined in the specification.

Success Criteria

Plan and Design

Evaluation

Testing & Remedial Action

Development
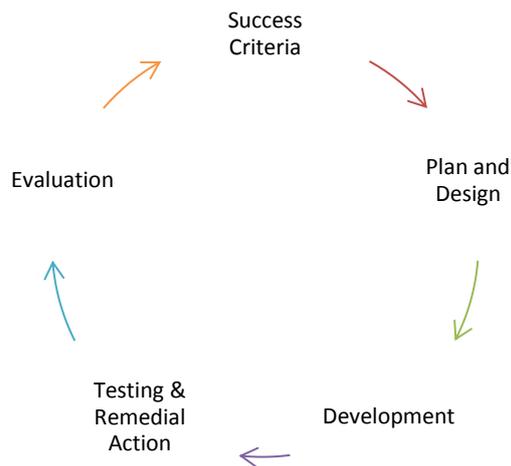
Learners will need to create suitable algorithms which will provide a solution to the problems identified in the project. They will then code their solution in a suitable programming language. The solution must be tested at each stage to ensure they solve the stated problem and learners must use a suitable test plan with appropriate test data.

2.1.2 Teacher preparation and Instructions for Programming Project

Learners will be expected to plan, design, develop, test and evaluate a simple coded program as part of the programming project completion. There will be no specialised hardware required for the programming project task completion by learners. The programming task can be solved using one of the following programming languages:

Drag and drop, for example:

Scratch https://scratch.mit.edu/
GameMaker http://www.yoyogames.com/gamemaker
Greenfoot http://www.greenfoot.org/door
Snap http://snap.berkeley.edu/ (Formerly BYOB)

Textually derived, for example:

Python https://www.python.org/ or https://www.codecademy.com/learn/python
Small Basic http://smallbasic.com/
Visual Basic / .NET https://msdn.microsoft.com/en-gb/library/2x7h1hfk.aspx
Java http://www.oracle.com/technetwork/java/index-138747.html
JavaScript http://docs.oracle.com/javase/6/docs/technotes/guides/scripting/programmer_guide/
C# or other C family language https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx

Before learners start a programming project, it is recommended that the teacher works through it themselves, so that they can anticipate any areas learners may find particularly challenging.

Specific Skills, Knowledge and Understanding:

Programming techniques:
Learners should have studied the following programming techniques before starting the programming project:
- Identify and use variables, operators, inputs, outputs and assignments
- Understand and use the three basic programming constructs used to control the flow of a program:
  - Sequence
  - Selection
    - IF Statements
  - Iteration
    - Count and condition controlled loops: WHILE and FOR
- Understand and use basic string manipulation
  - Concatenation only
- Use different types of data:
  - Integer
  - Boolean
  - Real numbers
  - Text
  - Character and string
- Define and use arrays (or equivalent) as appropriate when solving problems
  - one dimensional arrays (or similar)
- Use the common Arithmetic operators
- Use the common Boolean operators

How to decide on their success criteria:

Learners will need to read and understand each part of the task from the project to be able to state 'what the solution at that point should do' while creating their success criteria. An example is shared below:

e.g. "Extend your program to allow users to play up to 5 turns."

Success Criteria:

1. Create code to allow 5 turns to be played

2. Let the user know what turn they are on

3. Ask user if they want to play another turn

4. Ensure that the user types Yes or No

5. Ensure that 5 turns at most are allowed

How to plan a programming project:

Learners need to create a plan for the programming project in order to meet their success criteria. This could be done using flowchart, pseudocode or a combination of both. It must be a design. It is not acceptable to present the completed solution or reverse engineered code as a plan.

How to develop a solution using programming language:

Learners choose a programming language from the earlier list to develop a coded solution to the task. They must evidence iterative development of their codes used to get their final solution and add comments where appropriate.

An example of how to insert comment in Scratch programming language is done by right clicking on the work area and selecting add comment –

There are other ways to insert comments within different programming languages if learners choose to use another programming language as listed above.

Learners do not need to screen shot every line of code they create – but rather sections of code. They should focus on describing the steps they took in general, but also highlighting any unique points their code may have.

As learners progress through the task, it is likely that their final code will develop from their designs, either due to realisation of mistakes in the design, or decisions to make their code more efficient, etc. Showing this development is important – learners should be awarded credit in the design section, where they show re-design of earlier plans as they progress. The same applies later for testing.

How to test and evaluate:

Learners will test if the program works. While testing at this level is not expected to be exhaustive, it should be enough to ascertain that the code works for any expected situations / outcomes for the solution.

Most sections require candidates to test 'on the fly'. Some sections may require planning of a test table/section. Again, where candidates plan testing, and then adapt this later to be more thorough or due to edits in the designs themselves, credit should be awarded.

Evidence of programs running / solution work may be submitted using video or screenshots. All evidence must be submitted electronically.

Assessment / Marking of Programming project:

The programming project is assessed in the same way, whether it is solved using a textually derived programming language, or a drag and drop programming language. It is important to keep in mind that not every learner will be able to independently to produce a full working solution to the programming project. Near working and partial working solutions still generate credit within different mark bands, as described in the mark scheme.

There is no weighting placed on the efficiency of the program. If learners have used the listed constructs to achieve the specified outcome, then they meet the criteria, regardless of efficiency.

Some learners are likely to require more support than others. The level of support will determine the mark band awarded. Where learners have received support, this should be noted in the Unit Recording Sheet (URS) and marks adjusted accordingly.

If a learner requires support in doing part of the task, and then demonstrates the same skill / programming technique later on with limited support, then they will gain more credit than a learner who needed significant support throughout the programming project.

Guidance on the production of the Programming project:

It is recommended that evidence is produced over several sessions, which may focus on a specific part of the programming project. These may be interspersed with opportunities to learn computer science related knowledge and develop appropriate practical skills.

Each candidate must produce individual and authentic evidence for the task. Centres may give support and guidance to candidates.

This support and guidance should focus on checking that candidates understand what is expected of them. It is not acceptable for centres to provide model or 'similar-but-different' solutions to the programming project.

In general, candidates must be guided on the use of information from other sources to ensure that confidentiality and intellectual property rights are maintained at all times. It is essential that any material used directly from a source is appropriately and rigorously referenced.

Candidates must observe certain procedures in the production of assessments:

- Screenshots, tables and graphs may be produced using appropriate ICT tools. These should be inserted into the written work at the appropriate place
- Any copied material must be suitably acknowledged
- Quotations must be clearly marked and a reference provided wherever possible

Work submitted for moderation or marking must be marked with the:

- centre number
- candidate number
- candidate name
- unit code and title
- assignment title

Work submitted in digital format (CD, DVD, USB Pen Drive or online) must be in a suitable file structure as detailed in Appendix 5e 'files' of the specification.

3 Assessment and marking of written tests and programming task

3.1 Forms of assessment

OCR's Entry Level Certificate in Computer Science is a single entry qualification consisting of three parts that are assessed by the centre and externally moderated by OCR. Learners must complete all three sections of the content of the qualification.

The starting point for marking the Assessment tests and the Programming Project is the mark scheme issued by OCR. These contain levels of criteria for the skills, knowledge and understanding that the candidate is required to demonstrate.

The assessment of the internally assessed task should be marked by the teacher according to the given marking criteria for the relevant task using a 'best fit' approach. For each of the assessment objectives, the descriptor provided in the marking criteria that most closely describes the quality of the work being marked should be selected.

Marking should be positive, rewarding achievement rather than penalising failure or omissions. The awarding of marks **must be** directly related to the marking criteria.

Teachers should use their professional judgement in selecting band descriptors that best describe the work of the learner to place them in the appropriate band for each assessment objective section. If there is any contradiction found between the Handbook and the specification it is the specification that takes precedence.

To select the most appropriate mark in the band descriptor, teachers should use the following guidance:

- where the learner's work convincingly meets the statement, a suitable 'best fit' mark within the highest mark band should be awarded

- where the learner's work adequately meets the statement, a suitable 'best fit' mark within the middle mark band should be awarded

- where the learner's work just meets the statement, a suitable 'best fit' mark within the lowest mark band should be awarded.

Teachers should use the full range of marks available to them and award full marks in any band for work which fully meets that descriptor.

There should be clear evidence that work has been attempted and some work produced. If a learner submits no work for a section of the content, then the learner should be indicated as being absent from that section. If a learner completes any work at all for the sections of the subject content, then the work should be assessed according to the marking criteria and the appropriate mark awarded, which may be zero.

The OCR set Programming Projects will be available on OCR's Interchange. Guidance on how to access these projects from Interchange is available on the OCR website at http://www.ocr.org.uk/ocr-for-teachers/teachers-beta/interchange/interchange-step-by-step-guides/. Centres must ensure that learners undertake a suitable project to use a programming language which allows full access to the programming techniques described within the specification. The Programming Projects will be set by OCR for centres to choose from for their Entry Level Certificate in Computer Science qualification.

There are 3 Assessment Objectives in OCR Entry Level Certificate in Computer Science. These are detailed in the table below. Learners are expected to:

	Assessment Objective
AO1	Demonstrate Knowledge and Understanding Learners demonstrate their ability to recall, select and communicate their knowledge and understanding of concepts, issues and terminology.
AO2	Apply Knowledge and Understanding Learners demonstrate their ability to apply their knowledge and understanding.
AO3	Analyse, Evaluate and Apply Learners demonstrate their ability to analyse problems in computational terms to make reasoned judgement and to design, program, and evaluate solutions.

Success Criteria (0–3 marks available)

	Marks		
Marking criteria	**1**	**2**	**3**
AO1 – 0 AO2 – 3 AO3 – 0	• There is an attempt to identify some success criteria, but these only cover part of the solution and are incomplete	• There is an attempt to identify most success criteria adequately, and these relate to the majority of the requirements listed	• There is an attempt to convincingly identify the majority of success criteria, and these relate to the requirements listed

0 marks = no response or no response worthy of credit.

Planning and Design (0–6 marks available)

	Marks		
Marking criteria	**1–2**	**3–4**	**5–6**
AO1 – 0 AO2 – 2 AO3 – 4	• There is a statement(s) of what the intended program will do. This may not always reflect what the task requires • There is an attempt to create a flow chart for the program, but this is incomplete or appears non-functional • There are some tests given, but these are basic and use normal data only • There is an attempt to identify any Input, Output and Processing needs	• The learner has outlined how their program will work and this adequately matches any success criteria given • There is a flow chart produced that adequately maps a working solution to the problem, although it may contain some errors • There are a range of tests suggested using normal and erroneous data, but do not cover the entire solution proposed • Input, Output and Processing needs are identified and adequately meet the solution, although may not be complete	• The learner has described how their program will work and this convincingly matches the needs of the task • There is an accurate flow chart representing the proposed solutions that convincingly produces a functional solution • There are normal and erroneous tests for all parts of the solution as needed and the tests would provide convincing evidence that the solution is effective • Input, Output and Processing needs are clearly identified and cover all areas of the solution

0 = no response or no response worthy of credit.

Development (0–5 marks available)

Marking criteria	Marks		
	1	**2–3**	**4–5**

AO1 – 0
 	
AO2 – 0
 	
AO3 – 5
	• There is evidence of some of the following techniques:
	
	  • Input
	  • Output
	  • Data Storage
	  • Selection
	  • Iteration
	  • Arithmetic Operators
	  • Comments
	
	• There is little evidence of the development of the program, which will be limited and may not fully describe the steps taken to reach a solution
	
	• The explanations of the code leave doubt as to the understanding of the techniques used
		• There is evidence of a range of the following techniques, which may not always be used efficiently:
		
		  • Input
		  • Output
		  • Data Storage
		  • Selection
		  • Iteration
		  • Arithmetic Operators
		  • Comments
		
		• There is adequate evidence showing the development of the solution, but this may contain omissions
		
		• The explanations adequately support the learners understanding of the techniques used
			• There is convincing evidence of a range of the following techniques, which are generally used efficiently:
			
			  • Input
			  • Output
			  • Data Storage
			  • Selection
			  • Iteration
			  • Arithmetic Operators
			  • Comments
			
			• There is convincing evidence of the development of the solution and it provides a full narrative of the process
			
			• Explanations convincingly explain the learners understanding of techniques used

0 = no response or no response worthy of credit.

Testing and Remedial Actions (0–3 marks available)

Marking criteria	Marks		
	1	2	3
AO1 – 0 AO2 – 1 AO3 – 2	• The results of limited tests are evidenced with respect to success or failure • There is no or little evidence given to show an attempt to correct errors that are found in the solution	• The results of most tests are evidenced with respect to success or failure • There is some evidence given to show that errors have been adequately solved and re-tested	• The results of all tests are evidenced with respect to success for failure • There is convincing evidence that errors have been corrected and the program is functional

0 = no response or no response worthy of credit.

Evaluation (0–3 marks available)

Marking criteria	Marks		
	1	2	3
AO1 – 2 AO2 – 0 AO3 – 1	• There are limited statements about whether the solution has been successful • The link between evidence of testing and Success Criteria is weak and vague	• There are some statements that adequately review the success of the project • The link between evidence of testing and Success Criteria is adequate	• There is a full coverage of statements to reflect the Success Criteria • The link between evidence of testing and Success Criteria is convincing and covers all Success Criteria

0 = no response or no response worthy of credit

3.2 Guidance on external moderation of internal marking

The purpose of moderation is to bring the marking of internally-assessed components in all participating centres to an agreed standard. This is achieved by checking a sample of each centre's marking of learner's work. Please make sure that learners and centre have declared that the work submitted for assessment is the learner's own work by completing a centre authentication for (CCS160_CS). Please see Section 4c of the specification for further details.

Following internal standardisation, centres submit marks to OCR and the moderator. If there are fewer than 10 learners, all the work should be submitted for moderation at the same time as marks are submitted.

Once marks have been submitted to OCR and your moderator, centres will receive a moderation sample request. Samples will include work from across the range of attainment of the learners' work.

There are two ways to submit a sample:

Moderation via the OCR Repository – Where you upload electronic copies of the work included in the sample to the OCR Repository and your moderator accesses the work from there.

Postal moderation – Where you post the sample of work to the moderator electronically using CD, DVD or USB Pen Drive. No paper based sample of work will be accepted for this qualification.

The method that will be used to submit the moderation sample must be specified when making entries. The relevant entry codes are given in Section 4a of the specification.

All learners' work must be submitted using the same entry option. It is not possible for centres to offer both options within the same series.

Centres will receive the outcome of moderation when the provisional results are issued. This will include:

Moderation Adjustments Report – Listing any scaling that has been applied to internally assessed components.

Moderator Report to Centres – A brief report by the moderator on the internal assessment of learners' work is sent to centre.

Command Words

The command words below will be used consistently in all assessment materials and resources.

1–3 marks
State - Give a specific name, value or other brief answer without explanation or calculation.
Calculate - Obtain a numerical answer showing the relevant stages in the working.
Identify - Provide an answer from a number of possibilities. Recognize and state briefly a distinguishing factor or feature.
Draw - Produce (a picture or diagram) by making lines and marks on paper with a pencil, pen, etc.
Add - Join something to something else so as to increase the size, number, or amount.
Define - Give the precise meaning of a word, phrase, concept or physical quantity.
Outline - Give a brief account or summary.

4–5 marks
Explain - Give a detailed account including reasons or causes.
Justify - Give valid reasons or evidence to support an answer or conclusion.
Analyse - Break down in order to bring out the essential elements or structure. To identify parts and relationships, and to interpret information to reach conclusions.
Compare - Give an account of the similarities and differences between two (or more) items or situations, referring to both (all) of them throughout.
Design - Produce a plan, simulation or model.

6 or more marks
Evaluate - To assess the implications and limitations; to make judgments about the ideas, works, solutions or methods in relation to selected criteria.
Discuss - (**LOR**) Offer a considered and balanced review that includes a range of arguments, factors or hypotheses. Opinions or conclusions should be presented clearly and supported by appropriate evidence.

Other – can be any number of marks

Write/ Rewrite Mark (letters, words, or other symbols) on a surface, typically paper, with a pen, pencil, or similar implement/ Write (something) again so as to alter or improve it.

Tick Mark (an item) with a tick or select (a box) on a form, questionnaire, etc. to indicate that something has been chosen.

Complete Provide all the necessary or appropriate parts.

Annotate Add brief notes to a diagram or graph.

Label Add title, labels or brief explanation(s) to a diagram or graph.

How In what way or manner; by what means. *Ideally – 'how much'/ 'what is' – should be preceded by a command word. E.g. 'identify how much' 'state what is'. However, in practice this language is not always accessible to GCSE level candidates and so words which are not strictly command words can be used. Please note only use those in the above list.*

List Give a sequence of brief answers with no explanation.

Show Give steps in a derivation or calculation.

Solve Obtain the answer(s) using algebraic and/or numerical and/or graphical methods.

Pseudocode, Boolean logic and Flowcharts

The following guide enables teachers to introduce learners on using pseudocode. Learners are not expected to memorise the syntax of this pseudocode and, when asked, may provide answers in any style of pseudocode they choose providing its meaning could be reasonably inferred by a competent programmer.

The guide below shows languages and Boolean logic that will be used in the internal assessments and indicates the limits and scope of each. Centres are free to go beyond these parameters.

Variables and constants

Variables and constants are assigned using the = operator.

```
x=3
```

```
name="Bob"
```

Variables and constants are declared the first time a value is assigned. They assume the data type of the value they are given.

Variables and constants that are declared inside a function or procedure are local to that subroutine.

Variables in the main program can be made global with the keyword `global`.

```
global userid = 123
```

Variables in the main program can be made constant with the keyword const.

```
const vat = 20
```

Outputting to screen

```
print(string)
```

```
print(variable)
```

Example

```
print("hello")
```

```
print(myAge)
```

Taking Input from User

```
variable=input(prompt to user)
```

Example

```
name=input("Please enter your name")
```

Iteration – count-controlled

```
for i=0 to 7

    print("Hello")

next i
```

Will print hello 8 times (0-7 inclusive)

Iteration – condition-controlled

```
while answer!="computer"

     answer=input("What is the password?")

endwhile

do

     answer=input("What is the password?")

until answer=="computer"
```

Logical operators

```
AND OR NOT
```

e.g.

```
while x<=5 AND flag==false
```

Comparison operators

==	Equal to	e.g. a==b
!=	Not equal to	e.g. a!=b
<	Less than	e.g. a<b
>	Greater than	e.g. a>b

Arithmetic operators

+	Add	e.g. x=7+9 gives 16
-	Subtract	e.g. x=8-3 gives 5
*	Multiply	e.g. x=8*5 gives 40
/	Divide	e.g. x=10/2 gives 5

Selection

Selection will be carried out with if/else.

```
if/else

if entry=="a" then
     print("You selected A")
elseif entry=="b" then
     print("You selected B")
else
     print("Unrecognised selection")
endif
```

Arrays

Arrays will be 0 based and declared with the keyword `array`.

```
array names[5]

names[0]="Ahmad"

names[1]="Ben"

names[2]="Catherine"

names[3]="Dana"

names[4]="Elijah"

print(names[3])
```

Comments

Comments are denoted by //

`print("Hello World")` //This is a comment about hello world

Boolean algebra

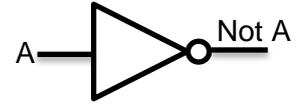When Boolean algebra is used in questions, the notation described below will be used.
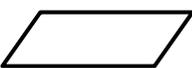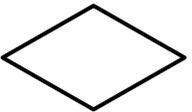
AND – Conjunction

A — A AND B
B

A	B	A AND B
T	T	T
T	F	F
F	T	F
F	F	F

OR – Disjunction

A — A OR B
B

A	B	A OR B
T	T	T
T	F	T
F	T	T
F	F	F

NOT – Negation

A — Not A

A	Not A
T	F
F	T

Flowcharts

Flow charts like pseudocode are informal but the most common flowchart shapes are:

	Line	An arrow represents control passing between the connected shapes
	Terminator	This oval shape represents the "Start" and "End" of the process.
	Process	This rectangle shape represents something being performed or done.
	Input / Output	This parallelogram shape represents the input or output of something into or out of the flow chart.
	Decision	This diamond shape represents a decision (Yes/No or True/False) that results in two lines representing the different possible outcomes.

Resources

There are a huge number of resources on the web and these, of course, are changing all the time. In particular, teachers are reminded that groups such as CAS (Computers at School) http://community.computingatschool.org.uk/door and the American CSTA (Computer Science Teachers Association) http://www.csteachers.org/ are active in putting members in touch with up to the minute and exciting resources to back up the teaching of what is perhaps the most vibrant subject available to students.

Raspberry Pi:
The official Raspberry Pi website has forums and updates on developments. It is also a great place to see what people have been creating with the Raspberry Pi www.raspberrypi.org
We have worked in collaboration with Raspberry Pi and with leading practitioners to create resources to support the use of the Raspberry Pi in the classroom. You will find tutorials, a series of Classroom Challenges, a Resources Link detailing suitable external resources relating to the Raspberry Pi, plus additional items on our website. www.ocr.org.uk/raspberrypi

Cambridge OCR GCSE Computer Science MOOC:
GCSE Computer Science MOOC (Massive Open Online Course) is an open FREE online computer science learning gateway with lots of videos and activity sheets available to download to teachers and learners http://www.cambridgegcsecomputing.org/

Cambridge University Press (CUP):
CUP is the official publishing partners for OCR and have developed an online learning and textbook to support the GCSE Computer Science specification. It contains a range of resources covering all aspects of the specification http://www.cambridge.org/gb/education

Hodder Dynamic Learning:
Hodder is the official publishing partners for OCR and have developed a dynamic learning website to support the GCSE Computing specification. It contains a range of resources covering all aspects of the specification http://www.dynamic-learning.co.uk

Little Man Computer:
Just one of several online working demonstrations of how memory and the processor interact http://www.yorku.ca/sychen/research/LMC/

BBC Online Teaching and Learning Resources:
BBC has produced a lot of resources to help learners with developing their computational thinking and programming at all Key Stages http://www.bbc.co.uk/education/subjects/z34k7ty

Industry Publications:
There are many industry magazines which put computing into a work related context. As well as many articles and features on contemporary issues, it is useful to look at the jobs sections to see what skills are actually in demand at the moment. http://www.computerweekly.com/Home/
http://www.computing.co.uk/

Bletchley Park Museum:
A visit to the Bletchley Park Museum always provides a stimulating background to the history and importance of computing http://www.bletchleypark.org.uk/

Dick Baldwin:
Programming tutorials for various languages including some excellent SCRATCH tutorials
http://www.dickbaldwin.com/toc.htm

GameMaker:
http://www.yoyogames.com/gamemaker

Greenfoot:
Environment for introducing object oriented programming http://www.greenfoot.org/index.html

Snap (BYOB): Snap is a visual drag-and-drop programming language http://snap.berkeley.edu/

RoboMind:
Programmable robot environment http://www.robomind.net/en/index.html

Scratch:
Great resource from MIT for introducing programming that uses colour coded snap together blocks to create sequences of instructions. A good starting point for developing the logical programming processes not hindered by the syntax barrier http://scratch.mit.edu/
http://info.scratch.mit.edu/Educators

Python:
Python is a programming language that lets you work more quickly and integrate your systems more effectively. Python is an easy to learn programming environment https://www.python.org/
https://www.codecademy.com/learn/python

For further resources to support teaching Python:
http://www.learnpython.org/
http://docs.python-guide.org/en/latest/intro/learning/

Small Basic:
A simple 3rd generation BASIC http://smallbasic.com/
https://www.visualstudio.com/products/visual-studio-express-vs

Visual Basic / .net:
Standard free visual.net environment programming language and support
https://msdn.microsoft.com/en-gb/library/2x7h1hfk.aspx

Java:
http://www.oracle.com/technetwork/java/index-138747.html

JavaScript:
http://docs.oracle.com/javase/6/docs/technotes/guides/scripting/programmer_guide/

C# or other C family language:
https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx

Frequently Asked Questions (FAQ)

Programming
Do I have to use the Programming Projects issued by OCR?
Yes, you must use one of the Programming Projects released by OCR through Interchange.

Can I use any Programming Language?
The choice of language is entirely up to teacher, providing it is capable of solving the programming project task in a way that meets the assessment criteria. Please use drag-and-drop or textually derived Programming language to solve the Programming project. It is listed in Section 2d of the specification and Section 2.1 within the Teachers' Handbook.

Can candidates work in groups for this task?
No, candidates are expected to tackle the programming project individually.

Is this controlled assessment?
No, but teachers must maintain the integrity of the assessment for Entry Level Certificate in Computer Science.