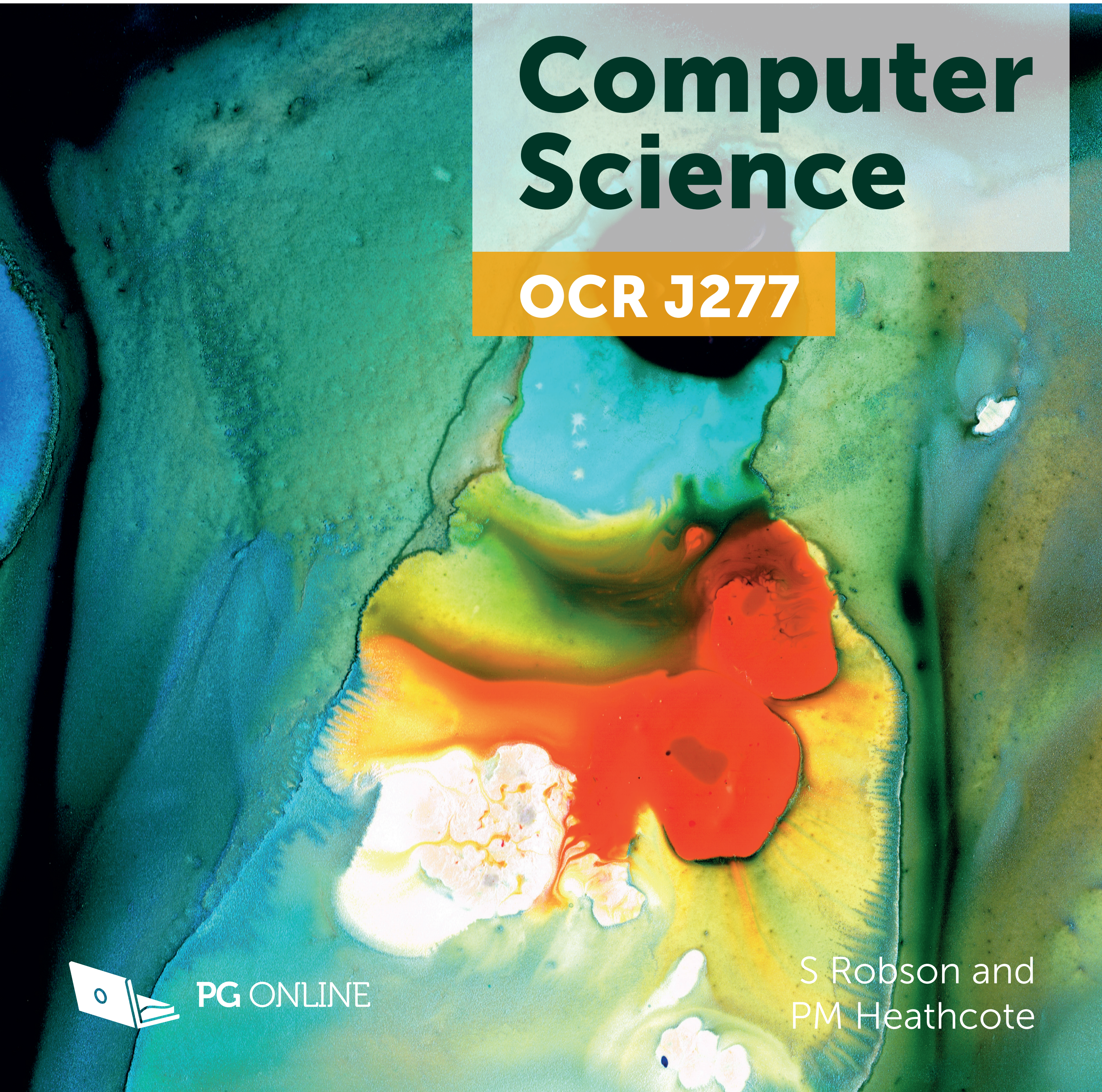
OCR
Oxford Cambridge and RSA
An OCR endorsed textbook

GCSE (9-1)

Computer Science

OCR J277



PG ONLINE

S Robson and
PM Heathcote

Contents

Paper 1 – Computer systems

Section 5
Ethical, legal, cultural and environmental impacts of digital technology 67

Paper 2 – Computational thinking, algorithms and programming

Section 6
Algorithms 81

Section 7
Programming 105

Section 8
Logic and languages 131

Paper 1 – Computer systems

Section 1

Systems architecture, memory and storage

1

Objectives

- Describe the architecture of the CPU
- Explain the purpose of the CPU as fetching and executing instructions stored in memory
- Describe what actions occur at each stage of the fetch-execute cycle
- Describe common CPU components and their function: ALU (Arithmetic Logic Unit), CU (Control Unit), cache, registers
- Explain the role and operation of the following CPU registers used in Von Neumann architecture: MAR (Memory Address Register), MDR (Memory Data Register), Program Counter, Accumulator
- Describe how common characteristics of CPUs affect their performance: clock speed, cache size, number of cores
- Explain the purpose and give typical characteristics and examples of embedded systems
- Explain the need for primary storage
    - o Describe the key characteristics and purpose of RAM and ROM
    - o Explain the need for virtual memory
- Describe the need for secondary storage including optical, magnetic and solid state storage
- Compare advantages and disadvantages between each type of storage device/medium
- Discuss data capacity of storage devices and calculate data capacity requirements
- Select suitable storage devices and storage media for a given application and the advantages and disadvantages of these, using characteristics: capacity, speed, portability, durability, reliability, cost

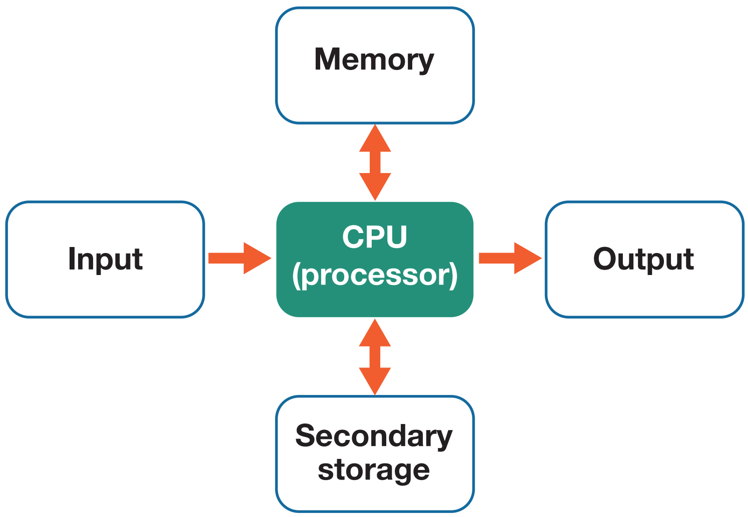
1.1 – Architecture of the CPU

Basic computer system model

A computer system is made up of hardware and software. Hardware is any physical component that makes up a computer. Software is any program that runs on a computer.

Computer systems are all around us. They are not just the PCs on a desk but include mobile phones, cash machines, supermarket tills and the engine management systems in a modern-day car.

The diagram below shows the basic model of a computer system.



All computer systems must have a **central processing unit (CPU)**, also called simply the **processor**, and at least one **input device** that gets data from the real world. This could be a mouse and keyboard on a conventional PC, a temperature sensor (thermistor) in a commercial greenhouse or the microphone on a mobile phone.

Input devices take real world data and convert it into a form that can be stored on the computer. The input from these devices is processed and the computer system will generate outputs. The **output device** could be, for example, a conventional computer screen, an actuator that opens or closes a greenhouse window, or the speaker that produces sound on a phone.

The computer must have **memory** (**primary storage**), used for holding instructions currently being executed and data that is being used.

Any computer system will have these four basic components.

The fifth component is **secondary storage**. The computer system may need to use stored data to perform the processing and, as a result of processing input, may generate data that is then stored. Storage devices such as hard disks can hold large amounts of data including databases, text documents, programs, music files and photographs.

Q1 Name **three** input, output and storage devices.

2.2 – Binary arithmetic and hexadecimal

Addition of binary numbers

Adding binary works in exactly the same way as adding denary numbers except this time you carry groups of 2 instead of groups of 10:

Adding in denary

```
12345
  134 +
-----
12479
```

$$\begin{array}{r} 7\,8\,2\,3\,5 \\ {}_{1}9_{1}7\ + \\ \hline 7\,8\,3\,3\,2 \end{array}$$

Notice that you carry 1 when you get to ten in a column so 5+7=12, write 2 in that column but carry 1 group of ten.

Adding in binary

```
10010
  101 +
-----
10111
```

$$\begin{array}{r} 1\,0\,0\,1\,1 \\ {}_{1}1_{1}1_{1}1\ + \\ \hline 1\,1\,0\,1\,0 \end{array}$$

Notice that: 0 + 0 = 0

0 + 1 or 1 + 0 = 1

1 + 1 = 0 carry 1

1 + 1 + 1 = 1 carry 1

Some more examples:

```
 10101100     00101101     00101101
 00010001+    10000101+    10000111+
---------    ---------    ---------
 10111101     10110010     10110100
```

Q6 Carry out the following binary number additions:

(a) 00110011 + 01000110

(b) 00010110 + 01110110

(c) 00001111 + 01110011

(d) 00101010 + 01111011

(e) 00011100 + 01110011

Overflow

The biggest number you can represent with 8 bits is 255 (i.e. 128+64+32+16+8+4+2+1).

If you add two binary numbers together that result in a number bigger than 255, it will need 9 or more bits. A computer stores things in memory in a finite amount of space. If you cannot represent the number in that amount of space because it is too big, then **overflow** occurs.

For example:

```
(252)  11111100
 (15)  00001111+
       ---------
(267) 100001011
```

The computer would need 9 bits to represent 267 so this 9th bit doesn't fit in the byte allocated. This is what is meant by an **overflow** error.

Lossy compression

Lossy compression is a data encoding method where files are compressed by removing some of the detail. For example, photographs can be stored using fewer colours so fewer bits are needed per pixel. This type of compression is used to compress images, audio files and video files, where it is easy to recognise an image or sound clip even if some data is missing.

A bitmap image (.bmp) is an uncompressed version of an image. If you save the same photograph as a JPEG file then it is still a high quality image with a colour depth of 24 bits but some of the data is lost where it is unlikely to be noticed.

Reduction of file size can also be achieved by reducing the colour depth from 24-bit colour, to 8-bit colour, for example. The human eye can tell the difference at this stage. You will see solid blocks of colour instead of gradual transitions in the photograph.

Here is a section of a photograph enlarged so you can see the difference:

24-bit colour

8-bit colour

Lossy compression formats are show below:

Type	File suffix	Compression Type	Explanation
JPEG	.jpg	Lossy	Good for photographs. Colour depth = 24 bits, RGB, 16.7 million different colours
Windows Media Player	.wmv	Lossy	Uses Windows Media compression
MP3	.mp3	Lossy	Audio files: Designed for downloading music from the Internet. In MP3 format you could fit 120 songs on a CD.
MPEG-1	.mpg	Lossy	Video files: Suitable for small low-resolution sequences on CD
MPEG-2	.mp2	Lossy	Video files: Suitable for full-screen, high resolution video on DVD

Exercises

1. (a) Add the following two 8-bit binary numbers. [2]

```
1 0 0 1 1 0 1 1
0 1 0 1 0 1 0 0
```

(b) An overflow error can occur when adding two 8-bit binary numbers.
Describe what is meant by an overflow error. [2]

OCR A451 June 2014 Qu 3

2. The number 73 could be a denary number or a hex number.

(a) If 73 is a hex number, calculate its value as a denary number.
You **must** show your working. [2]

(b) If 73 is a denary number, calculate its value as a hex number.
You **must** show your working. [2]

3. Numbers can be represented in denary, binary or hexadecimal.

(a) (i) Convert the binary number 01101001 to denary, showing your working. [2]

(ii) Convert the number 154 to binary. [2]

(b) The security code for an alarm system is a long binary number which begins

10001111100101111011 …

The technicians prefer to use hexadecimal to enter the security code.

(i) When the number is converted into hexadecimal, the first two digits are 8F as shown below.

Complete the gaps to show the next three digits.

Binary	1000	1111	1001	0111	1011
Hexadecimal	8	F	……..	……..	………

[3]

(ii) Explain why the technicians prefer to use hexadecimal. [2]

OCR A451 June 2013 Qu 5

4. (a) Explain why data is stored in computers in a binary format. [2]

(b) In the ASCII character set, the character codes for the first three capital letters are given below.

Letter	ASCII character code
A	0100 0001
B	0100 0010
C	0100 0011

(i) State how the ASCII character set is used to represent text in a computer. [2]

(ii) Convert the word CAB into binary using the ASCII character set. [2]

(iii) Explain why the ASCII character set is **not** suitable for representing text in all the languages of the world. [2]

OCR A451 Jan 2013 Qu 8

Encryption

Encryption is used primarily to protect data in case it has been hacked or accessed illegally. Data that is being transmitted over the Internet is vulnerable to hackers. For example, someone who uses an online shopping site will have to type in their payment details, such as a credit or debit card number, and it is essential that this information is kept secure. If they are paying by PayPal, they will have to type in their email address and password, which needs to be kept safe from anyone intercepting the transmission.

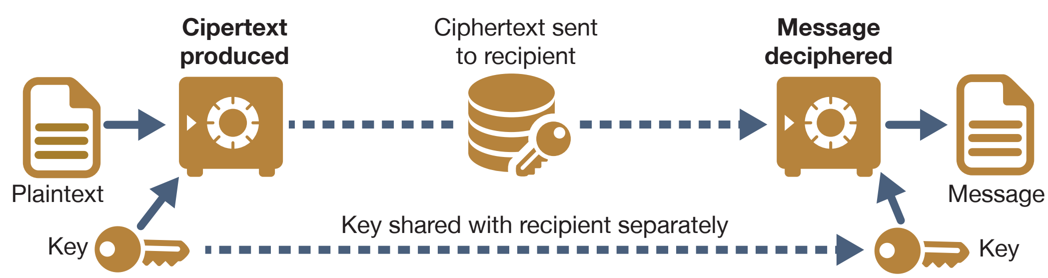
Whilst encryption won't prevent hacking, it makes the data incomprehensible unless the recipient has the necessary decryption tools.

Encryption terminology

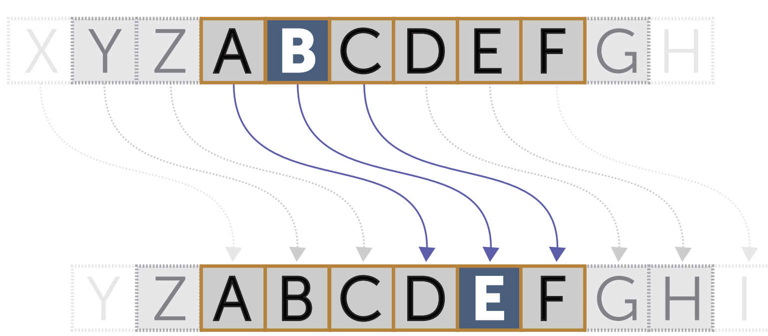
- **Plaintext**: the original message to be encrypted
- **Ciphertext**: the encrypted message
- **Encryption**: the process of converting plaintext into ciphertext
- **Key**: a sequence of letters, numbers and other characters used to encrypt or decrypt
- **Encryption algorithm**: the method for encrypting the plaintext

Symmetric encryption

Symmetric encryption uses a secret key which can be a combination of letters, numbers and other characters. A single key is used to encrypt and decrypt a message and must be given to the recipient of your message so that they can decrypt and read it.



A very simple example of symmetric encryption is the Caesar shift cipher, in which each letter is replaced by a letter n number of positions further on in the alphabet. The key in this case is 3:



Q2 Given the key 5, decode the encrypted message KNWJHWFHPJW BNQQ QTXJ

5.1 – Computer systems in the modern world

Computer technology impacts just about everything we do. Here are a few areas to consider.

Communication

We can keep in touch with friends and family all over the world through email, texting, phone calls and social networking sites. The latest news about world events can be spread instantly. We can find out whether a train is on time, what's on at the cinema, or exactly where a friend is at this moment, using our smartphones.

Can you imagine having to wait months for news of a family member who has emigrated to another country?

Employment

Computer technology has had a huge impact on employment. Many types of work have disappeared, and new jobs have been created.

Computer technology has already led to the loss of thousands of different jobs, for example in:

- clerical work
- manufacturing
- journalism

Robots could soon surpass humans in routine legal work, language translation and medical diagnosis – but plumbers, gardeners and physiotherapists will be hard to replace.

Thousands of new jobs have been created as a result of computer technology:

- software and hardware development
- creation of a multitude of new products from robots, 'smart homes' and mobile technologies to online learning materials and aids for disabled people

Q1 Name three jobs that you think could be computerised, and five jobs that cannot easily be computerised.

Shopping

Online shopping has given customers access to many different products that were traditionally difficult to purchase, as well as day to day items. This easy access has, in many towns, contributed to the closing of local and national stores leading to empty shops on the high street.

Example 4: Insertion sort

The same list of numbers is sorted into ascending order using an insertion sort:

9, 5, 4, 15, 3, 8, 11, 2

We leave the first item at the start of the list		**9**	**5**	**4**	**15**	**3**	**8**	**11**	**2**
5 is now inserted into the sorted list	**1st pass**	**5**	**9**	**4**	**15**	**3**	**8**	**11**	**2**
4 is now inserted into the sorted list	**2nd pass**	**4**	**5**	**9**	**15**	**3**	**8**	**11**	**2**
15 is now inserted into the sorted list (it stays where it is)	**3rd pass**	**4**	**5**	**9**	**15**	**3**	**8**	**11**	**2**
3 is now inserted into the sorted list	**4th pass**	**3**	**4**	**5**	**9**	**15**	**8**	**11**	**2**
8 is now inserted into the sorted list	**5th pass**	**3**	**4**	**5**	**8**	**9**	**15**	**11**	**2**
11 is now inserted into the sorted list	**6th pass**	**3**	**4**	**5**	**8**	**9**	**11**	**15**	**2**
2 is now inserted into the sorted list	**7th pass**	**2**	**3**	**4**	**5**	**8**	**9**	**11**	**15**

On each pass, the current data item is checked against those already in the sorted list (as shaded in the diagram). If the data item being compared in the sorted list is larger than the current data item, it is shifted to the right. This continues until we reach a data item in the sorted list which is smaller than the current data item.

For example, at the 5th pass 8 is compared with 15, and since 8 is smaller, 15 is shifted right.

8 is compared with 9, and 9 is shifted right.

8 is compared with 5, and as 8 is larger, it is inserted into the free space.

5th pass in summary:

8 is removed from the list temporarily	**3**	**4**	**5**	**9**	**15**		**11**	**2**
Since 15 > 8, it is now shifted to the right	**3**	**4**	**5**	**9**		**15**	**11**	**2**
Since 9 > 8, it is now shifted to the right	**3**	**4**	**5**		**9**	**15**	**11**	**2**
Since 5 < 8, 8 is now inserted into the sorted list	**3**	**4**	**5**	**8**	**9**	**15**	**40**	**53**

Q14 The following list of names is to be sorted into alphabetical sequence using an insertion sort. George, Jane, Miranda, Ahmed, Sophie, Bernie, Keith.

(a) What is the first name to be moved? What will the list look like after this name is moved?

(b) What is the second name to be moved? What will the list look like after this name has been moved?

(c) How many names have to be moved altogether before the list is sorted?

Example 8

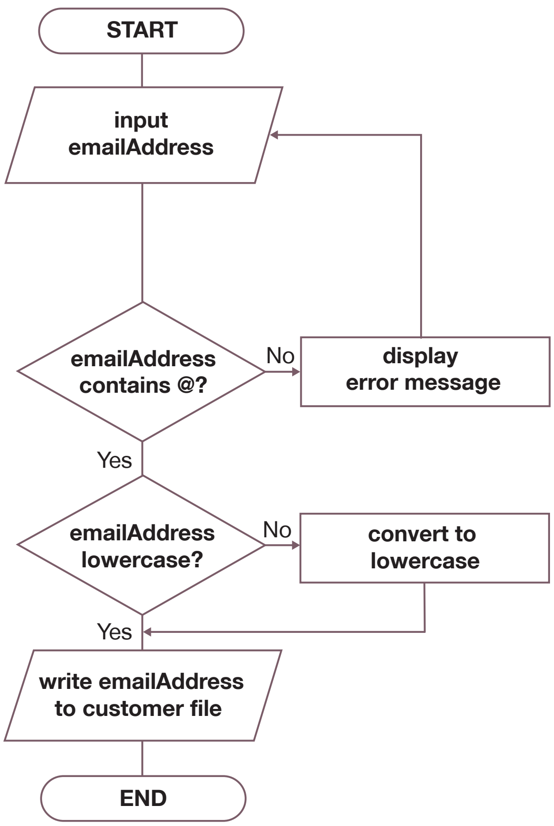
A computerised form prompts a user to enter their email address.

The validation rules check if the address has an @ symbol in it. If it doesn't, an error message is displayed, the text box is cleared and the system asks the user to enter the email address again. This continues until an appropriate address is entered.

The system then checks that the email address has been typed in lowercase and if not, it converts it to lowercase.

Once the email address is ok it is stored in the customer file.

The flowchart for this could be as follows:




```
emailAddress = input

while not hasAtSign
    print error message
    emailAddress = input
endwhile

if emailAddress is not
lowercase then
    Convert to lowercase
endif

Write emailAddress to
customer file
```

Q20 The **`if`** statement to check whether the address is lowercase is not needed. Modify the algorithm so that it performs the same task without the **`if`** statement.

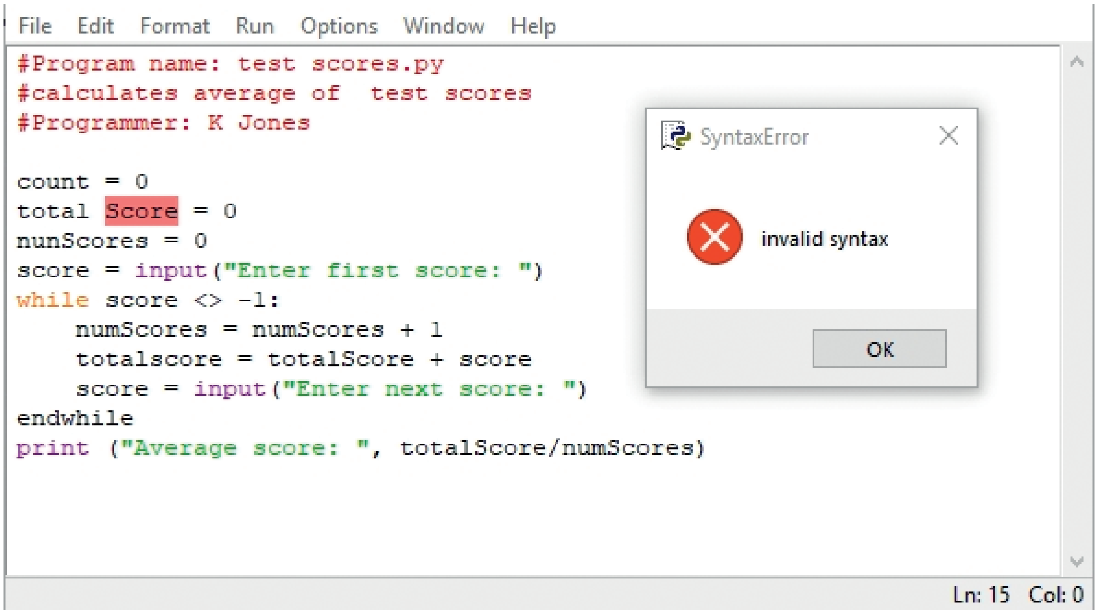
Q21 Write a pseudocode algorithm which inputs 10 numbers. Each time a number less than zero is input, the program displays which number it is, and its value. When all numbers have been input, display the average of all the negative numbers. Your algorithm should allow for the fact that there may be no negative numbers.

Sample output could be, for example:

```
Number 3   -8
Number 7   -20
Average of negative numbers = -14
```

- Auto-complete statements
- Colour-code key words such as **if, then, else**, comment statements, text within quotes etc.

The screenshot below shows a program that a user has entered, saved and attempted to run using the **Run** command at the top of the screen.



In addition, an IDE will also provide:

- Pretty printing
- Break points to stop the program running at certain points
- Watch windows to monitor changes in variable values

Q6 What is wrong with the statement? How does the IDE help identify comments, keywords, text in quote marks? Can you spot any other syntax errors?

Q7 List some other features of an IDE editor.

Error diagnostics

Once a program has been entered it can be saved and translated to machine code by an interpreter or a compiler. The interpreter or compiler will run error diagnostics on each line that the programmer has entered, when they try to run the saved program. The IDE will then allow them to correct any syntax errors, save the program, translate to object code using the interpreter or compiler and run it again. When the syntax errors have all been corrected and the program is run again, the IDE may discover and report a runtime error.

```
line 10, in <module>
    numScores = numScores + 1
NameError: name 'numScores' is not defined
>>>
```

Q8 Can you identify the problem? What must be amended to correct the program? There are more errors in the program – if you are a Python programmer you can try running the program.

Index

I

Index continued

I

PG ONLINE

OCR GCSE J277 (9-1) Specification map

	Paper 1: Computer systems	Section 1	Section 2	Section 3	Section 4	Section 5	Section 6	Section 7	Section 8
1.1	Systems architecture	✓							
1.2	Memory and storage	✓	✓						
1.3	Computer networks, connections and protocols			✓					
1.4	Network security				✓				
1.5	Systems software				✓				
1.6	Ethical, legal, cultural impacts of digital technology					✓			

	Paper 2: Computational thinking, algorithms and programming	Section 1	Section 2	Section 3	Section 4	Section 5	Section 6	Section 7	Section 8
2.1	Algorithms						✓		
2.2	Programming fundamentals							✓	
2.3	Producing robust programs								✓
2.4	Boolean logic								✓
2.5	Programming languages and IDEs								✓

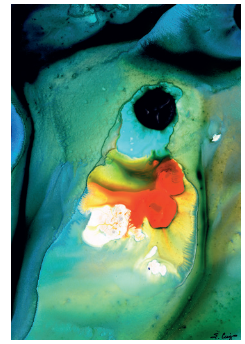
The content in each section of the textbook covers the same specification points as the corresponding downloadable teaching unit, e.g. Section 1 complements Unit 1.

Exclusively for teachers

To accompany each section in the textbook, there is a series of teaching units for the new OCR J277 (9-1) GCSE. Unit 5 pictured below is free.

OCR GCSE (9-1) J277

Computer Science



The aim of this book is to provide an accessible text for students, covering the OCR GCSE (9-1) Computer Science specification J277. It will be invaluable both as a course text and in revision for students nearing the end of the course. It is divided into eight sections, each broken down into manageable chapters of roughly one lesson.

Sections 6 and 7 of the textbook cover algorithms and programming concepts with a theoretical approach. This provides students with experience of writing, tracing and debugging pseudocode solutions without the aid of a computer. These sections would complement practical programming experience.

Each chapter contains in-text questions and exercises, some new and some from past examination papers, which can be set as homework. Answers to all these are available to teachers only, in a free Teachers' Supplement which can be ordered from our website **www.pgonline.co.uk**

About the authors

Susan Robson worked for International Computers Ltd after graduating from Manchester University with a degree in Computer Science. She spent the following 12 years in technical pre-sales for ECI Telecom, before moving into teaching. As a Head of Computer Science, she gained years of experience teaching GCSE and A Level Computing and has written successful textbooks and teaching materials. She is currently teaching Computer Science at King Alfred's Academy in Wantage.

Pat Heathcote is a well-known and successful author of Computer Science textbooks. She has spent many years as a teacher of A Level Computing courses with significant examining experience. She has also worked as a programmer and systems analyst, and was Managing Director of Payne-Gallway Publishers until 2005.

Cover picture:

'Love Song'
Acrylic on YUPO, 2012
© Sharon Cummings
www.sharoncummings.wordpress.com

This book has been endorsed by OCR.

ISBN 978-1-910523-21-6

9 781910 523216

PG ONLINE