

**Pearson Edexcel GCSE (9–1)**

**Computer Science**

**Student book Activity Answers**

**ISBN: 978 1 292 35999 1**

**First publication 2020**

## Table of contents

Table of contents.....	2
Topic 1 and 6: Computational thinking and problem solving.....	7
Activity 6 – Page 9 .....	7
Activity 7 – Page 13 .....	8
Exam-style questions – Page 17 .....	9
Activity 8 – Page 19 .....	10
Activity 9 – Page 19 .....	10
PRIMM activity – Page 22.....	11
Activity 10 – Page 23 .....	14
Exam-style questions – Page 23 .....	15
Activity 11 - Page 26 .....	16
Activity 12 - Page 29 .....	16
Activity 13 - Page 30 .....	17
PRIMM activity – Page 32.....	19
Exam-style questions – Page 33 .....	23
Activity 14 – Page 38 .....	26
Activity 15 – Page 40 .....	26
Activity 16 – Page 43 .....	27
Activity 17 – Page 43 .....	28
PRIMM activity - Page 43.....	29
Exam-style questions – Page 45 .....	34
Activity 18 – Page 49 .....	38
Activity 19 – Page 51 .....	38
Activity 20 – Page 52 .....	39
Activity 21 - Page 53 .....	40
Activity 22 - Page 55 .....	41
Activity 23 - Page 57 .....	43
Activity 24 - Page 60 .....	44
Activity 25 – Page 60 .....	45
PRIMM activity – Page 61.....	46
Exam-style questions – Page 62 .....	49
Activity 26 - Page 66 .....	53
Activity 27 – Page 68 .....	53

Activity 28 – Page 68 .....	54
Activity 29 – Page 70 .....	55
Activity 30 – page 74.....	56
PRIMM activity – Page 77 .....	57
Exam-style questions – Page 78 .....	61
Activity 31 – Page 83 .....	65
Activity 32 – Page 84 .....	70
Activity 33 – Page 89 .....	72
Activity 34 – Page 92 .....	72
Activity 35 – Page 93 .....	73
Primm activity – Page 95 .....	75
Exam-style questions – Page 97 .....	83
Activity 36 – Page 101 .....	86
Activity 37 - Page 102 .....	87
Worked example – Page 103.....	88
Activity 38 – Page 105 .....	89
Activity 39 – Page 107 .....	91
Activity 40 – Page 112 .....	91
Activity 41 – Page 113 .....	92
Activity 42 – Page 113 .....	92
Exam-style questions – Page 114 .....	94
Activity 43 – Page 119 .....	96
Activity 44 – Page 122 .....	96
Activity 45 – Page 125 .....	98
PRIMM activity – Page 126.....	100
Exam-style questions – Page 128 .....	110
Activity 46 – Page 131 .....	114
Activity 47 – Page 132 .....	115
Activity 48 – Page 133 .....	115
Activity 49 – Page 134 .....	117
Activity 50 – Page 137 .....	118
Activity 51 – Page 138 .....	118
Activity 52 – Page 140 .....	119
Activity 53 – Page 146 .....	120

Activity 54 – Page 146 .....	121
PRIMM activity – Page 147 .....	122
Exam-style questions – Page 149 .....	129
Activity 55 – Page 154 .....	134
Activity 56 – Page 155 .....	134
Activity 57 – Page 158 .....	136
Activity 58 – Page 159 .....	136
PRIMM activity – Page 160 .....	139
Exam-style questions – Page 163 .....	147
Activity 59 – Page 167 .....	150
Activity 60 – Page 167 .....	150
Activity 61 – Page 171 .....	151
Activity 62 – Page 173 .....	152
Exam-style questions – Page 175 .....	153
Topic 2: Data .....	156
Activity 1 – Page 179 .....	156
Activity 2 – Page 180 .....	156
Activity 3 – Page 181 .....	156
Activity 4 – Page 182 .....	156
Activity 5 – Page 183 .....	157
Activity 6 – Page 184 .....	157
Activity 7 – Page 186 .....	157
Activity 8 – Page 186 .....	158
Activity 9 – Page 187 .....	158
Activity 10 – Page 187 .....	159
Activity 11 – Page 189 .....	159
Activity 12 – Page 191 .....	159
Exam-style questions – Page 191 .....	161
Activity 13 – Page 193 .....	162
Activity 14 – Page 194 .....	162
Activity 15 – Page 195 .....	163
Activity 16 – Page 196 .....	163
Activity 17 – Page 200 .....	164
Activity 18 – Page 200 .....	164

Exam-style questions – Page 201 .....	165
Activity 19 – Page 202 .....	167
Activity 20 – Page 202 .....	167
Exam-style questions – Page 204 .....	168
Topic 3: Computers .....	169
Activity 1 – Page 209 .....	169
Activity 2 – Page 210 .....	170
Activity 3 – Page 213 .....	172
Activity 4 – Page 214 .....	174
Activity 5 – Page 214 .....	175
Exam-style questions – Page 217 .....	176
Activity 6 – Page 219 .....	178
Activity 7 – Page 219 .....	180
Activity 8 – Page 220 .....	181
Activity 9 – Page 223 .....	181
Activity 10 – Page 223 .....	181
Activity 11 – Page 225 .....	182
Exam-style questions – Page 226 .....	183
Activity 12 – Page 228 .....	185
Exam-style questions – Page 230 .....	187
Topic 4: Networks.....	188
Activity 1 – Page 232 .....	188
Activity 2 – Page 233 .....	189
Activity 3 – Page 234 .....	189
Activity 4 – Page 237 .....	190
Activity 5 – Page 239 .....	190
Activity 6 – Page 240 .....	191
Activity 7 – Page 244 .....	192
Activity 8 – Page 246 .....	195
Activity 9 – Page 247 .....	196
Activity 10 – Page 250 .....	197
Exam-style questions – Page 250 .....	199
Activity 11 – Page 253 .....	202
Activity 12 – Page 255 .....	202

Activity 13 – Page 256 .....	203
Exam-style questions – Page 258 .....	204
Topic 5: Issues and impact.....	206
Activity 1 – Page 262 .....	206
Activity 2 – Page 262 .....	207
Activity 3 – Page 263 .....	208
Activity 4 – Page 264 .....	209
Activity 5 – Page 266 .....	210
Exam-style questions – Page 266 .....	212
Activity 6 – Page 268 .....	214
Activity 7 – Page 269 .....	214
Activity 8 – Page 273 .....	214
Activity 9 – Page 276 .....	214
Activity 10 – Page 277 .....	215
Activity 11 – Page 277 .....	215
Activity 12 – Page 278 .....	215
Activity 13 – Page 279 .....	216
Activity 14 – Page 279 .....	216
Activity 15 – Page 279 .....	216
Exam-style question – Page 280 .....	217
Activity 16 – Page 282 .....	219
Activity 17 – Page 284 .....	220
Activity 18 – Page 284 .....	222
Activity 19 – Page 286 .....	223
Activity 20 – Page 288 .....	224
Activity 21 – Page 291 .....	224
Exam-style questions – Page 292 .....	225
Prepare for your exam – Paper 1 .....	227
Prepare for your exam – Paper 2 .....	228
Question 01 – Page 304 .....	228
Question 02 – Page 307 .....	229

## 1.1: Good practice: tools and strategies

### Introduction

All questions are given as in the student book for reference, with answers in blue type, including multiple choice questions, where the correct answer is shown in blue. Where program code is supplied as an answer, it is shown in coloured syntax, rather than in blue. Program code can be copied from this document and pasted into an editor.

The table of contents ties the textbook page for each activity to the page in this document where the solution can be found. Where a solution requires code, both the code in the question and the code for the solution are provided.

### Topic 1 and 6: Computational thinking and problem solving

The first five activities in the student book are all related to using an integrated development environment. As the choice of environment will affect the outcome of the activity, solutions have not been provided.

#### Activity 6 – Page 9

Here is a very short program.

```

1  # -----
2  # Global variables
3  # -----
4  score = 0
5  lives = 6
6
7  # -----
8  # Main program
9  # -----
10
11 while (score <= 50):
12     lives = lives + 2
13     score = score + 10
14 print ("Goodbye")

```

Use your IDE, breakpoints, stepping and variable inspection to answer these questions:

- What are the very first values of lives and score?  
lives = 6 and score = 0
- When lives is 12, what is the value of score?  
30
- When score is 40, what is the value of lives?  
14
- What are the very last values of lives and score?

## 1.1: Good practice: tools and strategies

`lives = 18 and score = 60`

- How many times does the value of score change?

`7 (nothing, 0, 20, 30, 40, 50, 60)`

### Activity 7 – Page 13

Here is a very short program.

```

1  # -----
2  # Global variables
3  # -----
4
5  num22 = 11
6  seats = 34
7  count = 0
8
9  # -----
10 # Main program
11 # -----
12
13 num22 = num22 +
14
15 while (seats > 0):
16 seats = seats - 10
17 print (seats)
18 # Should print 24, 14, 4, -6
19
20 seats = num2 + 12
21
22 count = seats - "5"
23 count = seats - 5
    
```

Use your IDE, breakpoints, stepping and variable inspection to find and correct:

- a syntax error

Line 13 needs a number added to it, such as 11 or num22

- an indentation error

Lines 16, 17, and 18 should be indented one level

- a name error

Line 20 uses a variable num2, but num2 does not exist. Change to num22.

- a type error.

Line 22 attempts to subtract a string from an integer. Change "5" to 5.



## 1.1: Good practice: tools and strategies

### Exam-style questions – Page 17

1. Here is an error message.

`IndexError: list index out of range`

Identify the type of error it is from the following options.

a) Runtime error

b) Syntax error

c) User error

d) Logic error (1 mark)

2. Give three features found in an Integrated Development Environment (IDE). (3 marks)

Editor, debugger, inspection, stepping and syntax checker variable

3. State the meaning of the term 'syntax' in a programming language. (1 mark)

The grammar/structure of the programming language

4. Explain why programmers use version control. (2 marks)

Keeping track of the changes to code is important because programmers may need to go back a level or two if they mess up their code.

## 1.2: Algorithms and programs

### Activity 8 – Page 19

Produce a written description of an algorithm for getting to school. It should start with leaving home and end with arriving at school.

Leave front door.

Walk down path to street.

Turn right.

Walk to end of the road.

Turn left and cross the road.

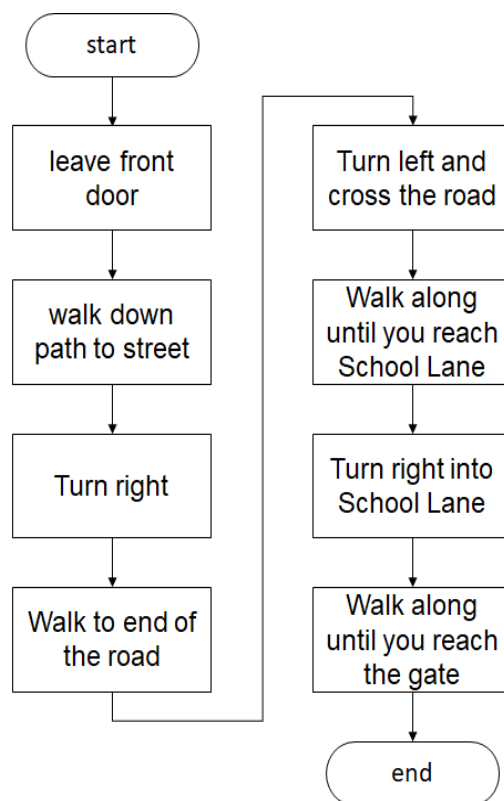
Walk along until you reach School Lane.

Turn right into School Lane.

Walk along until you reach the gate.

### Activity 9 – Page 19

Display the ‘journey to school’ algorithm, which you created in the previous activity, as a flowchart.



## 1.2: Algorithms and programs

### PRIMM activity – Page 22

This activity uses the code shown here. Use it for each of the steps shown.

```

1  # -----
2  # Global variables
3  # -----
4  b = 22
5  h = 44
6  a = 0
7
8  # -----
9  # Main program
10 # -----
11 a = (1/2) * b * h
12 print ("Area is", a)

```

#### Predict

1. What do you think the code will do?

Calculate the area of a triangle

2. What output do you think it will produce?

A sentence with some words and the area

#### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do? Did the output match your prediction? If not, how did it differ?

Yes, it matched.

#### Investigate

1. Remove the line 'b = 22'.
  - a. Run the code.
  - b. Carefully read the error message.

Programs/PRIMM/Area of a triangle.py", line 12, in <module>

```
a = (1/2) * b * h
```

NameError: name 'b' is not defined

- c. What is the message trying to tell you?

That the variable 'b' does not exist in the program

- d. Put the line back in.
2. Change the calculation to move the '(1/2)' between the 'b' and 'h'.
    - a. Run the code.
    - b. What does that tell you about the use of brackets?

Brackets are evaluated first

## 1.2: Algorithms and programs

- c. What does that tell you about the order of arithmetic operators?

Multiplication is commutative; order does not matter.

### Modify

Be sure to run the code after each change to check it still works.

1. Change the identifier names (variables) to be more meaningful. Remember to use camel case, if required.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  base = 22
5  height = 44
6  area = 0
7
8  # -----
9  # Main program
10 # -----
11 area = (1/2) * base * height
12 print ("Area is", area)

```

2. Using the existing output line, add lines to display the values of each variable before the total area.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  base = 22
5  height = 44
6  area = 0
7
8  # -----
9  # Main program
10 # -----
11 area = (1/2) * base * height
12 print ("Base is", base)
13 print ("Height is", height)
14 print ("Area is", area)

```

## 1.2: Algorithms and programs

### Make

A program is needed to calculate the price of an item after adding tax. The program must store the base price of an item, the tax rate (expressed as a decimal), and the final price. The program must output the base price, the tax rate, and the final price. The formula to calculate the final price is  $\text{base price} \times (1 + \text{tax rate})$ .

If the base price is 12.30 and the tax rate is 0.20, then the final price is 14.76.

Create a new program file named "Net\_Price.py". Lay out the sections of code as shown in the examples. Write and test your program.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  basePrice = 12.30
5  taxRate = 0.20          # 20 percent
6  total = 0.0
7
8  # -----
9  # Main program
10 # -----
11 total = basePrice * (1 + taxRate)
12 print ("Base price is", basePrice)
13 print ("Tax rate is", taxRate)
14 print ("Total price is", total)

```

## 1.2: Algorithms and programs

### Activity 10 – Page 23

Here is a written description of an algorithm.

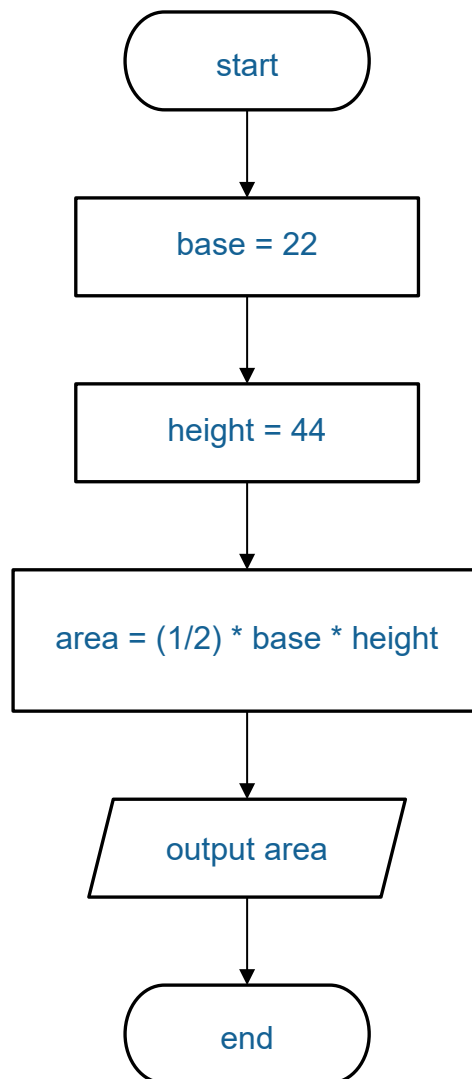
Set base of triangle to 22

Set height of triangle to 44

Set area of triangle to  $(1/2) * \text{base} * \text{height}$

Output area to display

Express this algorithm in a flowchart.

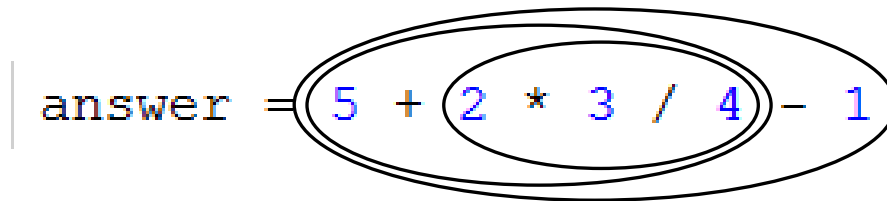


## 1.2: Algorithms and programs

### Exam-style questions – Page 23

- Here is the image of a line from a program. Amend the image with circles to show the order in which the values are evaluated, using the order of precedence rules.

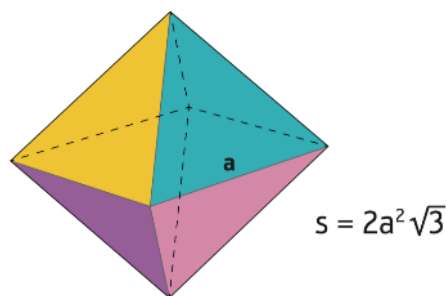
(3 marks)



- Compare the modulus (%) operator and the integer division (//) operator. (3 marks)

Both modulus and integer division take two inputs. They both divide (1) one input by the other. However, the modulus operator returns only the remainder (1) part of the result, while the integer division operator returns only the whole number (integer) (1) part of the result.

- The formula for calculating the surface area of an octahedron is shown in the image. Construct an expression, using the arithmetic operator symbols, to translate this formula into Python. (5 marks)



Remember, raising a number to the power of (1/2) is the same as taking the square root.

```
answer = 2 * (a ** 2) * (3 ** (1/2))
```

```
answer = 2 * (a * a) * (3 ** (1/2))
```

```
answer = 2 * a * a * 3 ** (1/2)
```

```
answer = 2 * a ** 2 * 3 ** (1/2)
```

```
marks (1)(1)(1)(1)(1)
```

## 1.3: Data types

### Activity 11 - Page 26

1. What other value could be assigned to the variable `finishedTest`?  
True
2. Would a value of `-15` be valid to assign to a variable that was declared as an integer?  
Yes, both negative and positive numbers, including zero, can be integers.
3. How do you know that the data type of the variable `cost` must be float?  
Currency has pence, which is a decimal number. The variable is initialised with a decimal number. In Python, decimal numbers are the float data type.
4. How do you know that the data type of the variable `colour` must be a string?  
The colour is set to green, so it is the name of the colours that must be used, such as 'red'. Words in quotation marks are strings.
5. What is the effect of adding more variable names, separated by commas, to the print statement?  
The value in the variables would be printed out on the line after the colour.
6. What is the purpose of the `#` character that appears on some lines?  
This is a comment to show notes added by the programmer to help explain the logic.

### Activity 12 - Page 29

Create a program file called 'My\_Data\_types.py'. Using the above example, write program code to meet these requirements.

1. Create four variables, one of each data type.
2. Choose meaningful identifier names, but different to the example, expressed in camel case.
3. Initialise the variables to sensible default values.
4. Output the content of each variable, followed by its data type, both on the same output line.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  isWeekday = True           # Initialisation as Boolean
5  max = 999                  # Initialisation as integer
6  temp = 21.5                # Initialisation as float
7  aStudent = "Aidan"        # Initialisation as string
8  yes = "Y"                  # Initialisation as character (string)
9
10 # -----
11 # Main program
12 # -----

```



## 1.3: Data types

```

13 print (isWeekday, type(isWeekday))
14 print (max, type(max))
15 print (temp, type(temp))
16 print (aStudent, type(aStudent))
17 print (yes, type(yes))

```

Console Output:

```

True <class 'bool'>
999 <class 'int'>
21.5 <class 'float'>
Aidan <class 'str'>
Y <class 'str'>

```

### Activity 13 - Page 30

```

1
2 # -----
3 # Global variables
4 # -----
5 myName = ""           # Empty string
6 heightMeters = 99.0   # Invalid height
7
8 # -----
9 # Main program
10 # -----
11 print (myName)
12 print ("This demonstrates the use of the input() function.")
13 myName = input ("Please type your name and press the enter key. ")
14 heightMeters = float(input("Please enter your height in meters. "))
15 print (myName, "is", heightMeters, "meters tall.")

```

Answer these questions based on the code above. You may want to load the code into your programming environment to investigate its behaviour.

1. If you printed the initial contents of `myName`, what would you expect to see on the display?

You would see nothing, as the string is empty. However, an empty line would be displayed because by default the `print()` function causes a line feed to be sent to the display.

2. Why should the variable `heightMetres` be a data type of float?

People are not rounded to the nearest metre, so their heights will include decimals, such as 1.6 metres.

3. Why does initialising the variable `heightMetres` to 99.0 mean it has a data type of float?

It has a decimal in the number. Without it, it would be initialised as an integer.

## 1.3: Data types

4. Why does the input line setting myName not use the str() function to convert the keyboard input to a string?

The keyboard returns characters, and people's names are characters, not numbers. Therefore, they are strings by default.

5. Construct the lines necessary to ask for the user's age in whole years and display the age in the output message.

### *Solution*

```
age = 0
age = int(input("Please enter your age in whole years. "))
print (myName, "is", heightMeters, "meters tall and is", age, "years old.")
```

## 1.3: Data types

### PRIMM activity – Page 32

This activity uses the code shown here.

```

1  # -----
2  # Global variables
3  # -----
4  firstName = ""
5  lastName = ""
6  theMonth = 0
7  theDecimal = 0.0
8  memberID = ""
9
10 # -----
11 # Main program
12 # -----
13
14 firstName = input("Enter your first name ")
15 lastName = input("Enter your last name ")
16 theMonth = int(input("Enter the number of your birth month (January is 1) "))
17 theDecimal = float(input("Enter a decimal number "))
18
19 # Create a memberID
20 memberID = firstName[0] + lastName[0] + str(100 - theMonth)
21 print(memberID)
22
23 # Create a memberID
24 memberID = firstName[0] + str(100 - theMonth) + lastName[len(lastName)-1]
25 print(memberID)
26
27 # Create a memberID
28 memberID = firstName[len(firstName)-1] + str(theDecimal * theMonth)
29 print(memberID)

```

### Predict

1. What do you think the code will do?

Generates different member id keys and prints them.

2. What output do you think it will produce?

Only the member ids are printed. They're made up of different combinations of letters from the name and numbers. No messages are included.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

## 1.3: Data types

Yes, it did.

- Did the output match your prediction? If not, how did it differ?

None.

### Investigate

- Why should `theMonth` be stored with a data type of integer?

Because it can be represented as whole numbers 1 to 12.

- Why should `theDecimal` be stored with a data type float?

Because it has a decimal in it with a fractional part.

- What data type would a variable named `windowOpen` best be stored as?

Boolean, because it can only be one of two values, opened or closed.

- What happens to the output if the user types in an integer when they should type in a decimal?

There is no change, because the input line will convert an integer input to a float, which has a decimal.

### Modify

- Currently the `memberID` is created by concatenation. After each print line, add a new line which prints each `memberID` element separated by a comma.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  firstName = ""
5  lastName = ""
6  theMonth = 0
7  theDecimal = 0.0
8  memberID = ""
9
10 # -----
11 # Main program
12 # -----
13
14 firstName = input("Enter your first name ")
15 lastName = input ("Enter your last name ")
16 theMonth = int(input("Enter the number of your birth month (January is 1) "))
17 theDecimal = float(input("Enter a decimal number "))
18
19 # Create a memberID
20 memberID = firstName[0] + lastName[0] + str(100 - theMonth)
21 print (memberID)
```

## 1.3: Data types

```

22 print (firstName[0], lastName[0], str(100 - theMonth))
23
24 # Create a memberID
25 memberID = firstName[0] + str(100 - theMonth) + lastName[len(lastName)-1]
26 print (memberID)
27 print( firstName[0], str(100 - theMonth), lastName[len(lastName)-1])
28
29 # Create a memberID
30 memberID = firstName[len(firstName)-1] + str(theDecimal * theMonth)
31 print (memberID)
32 print (firstName[len(firstName)-1], str(theDecimal * theMonth))

```

- Does this approach create an acceptable `memberID`? Why or why not?

No, because the member id will have spaces in it.

- What is the difference between using concatenation and using commas?

Commas introduce spaces between each variable, while concatenation runs them all together without spaces.

### Make

A program is required to generate IDs for school lockers. The school has corridors named red, blue, yellow and green. The lockers are in banks of 3 rows x 12 columns. A locker ID is made up of the first letter of the corridor name and a number generated by raising the row to the power of the column.

Corridor	Row	Column	Row <sup>Column</sup>	Locker ID
red	1	12	1 <sup>12</sup>	r1
blue	3	8	3 <sup>8</sup>	b6561
yellow	2	3	2 <sup>3</sup>	y8
green	2	10	2 <sup>10</sup>	g1024

- Display the type of a string variable.
- Display the type of an integer variable.
- Ask the user to enter a colour, a row, and a column.
- Construct and display a locker ID.

A sample console session is shown.

Console session:

corridor is <class 'str'>

row is <class 'int'>

Enter a colour (red, blue, yellow, green) *green*

Enter a row (1-3) *2*

Enter a column (1-3) *10*

location is 2green10

location is g1024

## 1.3: Data types

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  corridor = ""
5  row = 0
6  column = 0
7  location = ""
8
9  # -----
10 # Main program
11 # -----
12
13 # Display types for variables
14 print ("corridor is ", type(corridor))
15 print ("row is ", type(row))
16
17 corridor = input ("Enter a colour (red, blue, yellow, green) ")
18 row = int(input("Enter a row (1-3) "))
19 column = int(input("Enter a column (1-3) "))
20
21 location = str(row) + corridor + str(column)
22 print ("location is", location)
23
24 location = corridor[0] + str(row ** column)
25 print ("location is", location)

```

## 1.3: Data types

## Exam-style questions – Page 33

1. Complete the table to show data type, declaration, and initialisation for variables storing each value in the first column. (12 marks)

Value	Data type	Declaration	Initialisation
Body temperature	real	bodyTemp = float()	bodyTemp = 0.0
Whether the door is open or closed	Boolean	doorStatus = Boolean()	doorStatus = True
Number of students in a school	integer	numStudents = int()	numStudents = -1
Street name	string	streetName = str()	streetName = "Seaside Circle"

Award one mark for each correct cell, up to a maximum of 12.

Allow carry through errors in declaration and initialisation, if they match the data type given.

2. Here is part of a program.

```

1 # -----
2 # Global variables
3 # -----
4 theString = str()
5 theString = "EDUCATION"
6
7 # -----
8 # Main program
9 # -----
10 # ==> Add your code here

```

Amend the program to:

Display the length of the string.

Display all five of the vowels (a, e, i, o, u) using string indexing. (5 marks)

## Solution

```

1 # -----
2 # Global variables
3 # -----
4 theString = str()
5 theString = "EDUCATION"

```

## 1.3: Data types

```

6
7 # -----
8 # Main program
9 # -----
10 # =====> Add your code here
11 print (len(theString))
12 print (theString[0], theString[2], theString[4], theString[6], theString[7])

```

Award one mark for the use of len() function (1).

Award one mark for the use of correct indices (1) up to a maximum of 4.

3. Here is part of a program.

```

1 # -----
2 # Global variables
3 # -----
4 # =====> Add your variables here
5
6
7 # -----
8 # Main program
9 # -----
10 # =====> Add your code here

```

Here is a console session output

Please enter a decimal number: 1.23

Please enter a decimal number: 4.56

Float: 5.7899999999999999

Integer: 5

Amend the program to:

- Explicitly declare and initialise two variables of data type float.
- Ask the user to enter two decimal numbers.
- Print the word 'Float' concatenated to the sum of the two numbers.
- Print the word 'Integer' concatenated to the sum of the two numbers converted to integer. (10 marks)

**Solution**

```

1 # -----
2 # Global variables
3 # -----
4 # =====> Add your variables here
5 num1 = float()
6 num1 = 0.0
7
8 num2 = float()

```



## 1.3: Data types

```

9  num2 = 0.0
10
11  # -----
12  # Main program
13  # -----
14  # =====> Add your code here
15  num1 = float (input ("Please enter a decimal number: "))
16  num2 = float (input ("Please enter a decimal number: "))
17  print ("Float: " + str (num1 + num2))
18  print ("Integer: " + str (int (num1 + num2)))

```

Instructions say to use concatenation, so award use of '+' symbol, not commas.

Award marks for:

- declaration of either variable (1)
- initialisation of either variable (1)
- meaningful variable names (1)
- note: question asks for explicit declaration; if implicit, then award (1) mark only
- prompt for either number input (1)
- conversion of either number input to float (1)
- print 'Float' and 'Integer' with correct format of output (1)
- use of concatenation in all output (1)
- conversion of float to string (1)
- conversion of float to integer (1)
- conversion of integer to string (1)

## 1.4: Selection and relational operators

### Activity 14 – Page 38

Here is the code for part of a game. Pieces are moved based on the roll of a single die.

```

1  # -----
2  # Global variables
3  # -----
4  theRoll = 1
5
6  # -----
7  # Main program
8  # -----
9
10 if (theRoll >= 4):
11     print ("Move " + str (theRoll) +
12           " spaces forward")
13 else:
14     print ("Move " + str (theRoll - 1) +
15           " spaces forward")

```

Complete the table to show the output for these rolls of the die.

Roll	3	4	5
Output	Move 2 spaces forward	Move 4 spaces forward	Move 5 spaces forward

### Activity 15 – Page 40

A driving school uses this rule to estimate how many 1-hour lessons a learner will require.

- Only people over the age of 17 can take driving lessons.
- Every learner requires at least 20 lessons.
- Learners under the age of 25 require more lessons – two additional lessons for each year under 25.

Create a program that inputs a learner's age and calculates the number of driving lessons needed.

#### Solution

```

1  # -----
2  # Global variables
3  # -----
4  age = 0                                # Learner's age
5  baseLessons = 20                       # Everyone needs 20 lessons
6  extraLessons = 0                       # Only for under 22 years
7
8  # -----

```

## 1.4: Selection and relational operators

```

9  # Main program
10 # -----
11 age = int (input ("Enter your age "))
12
13 if (age < 17):
14     print ("You're too young")
15 elif (age < 25):
16     # Need extra Lessons as very young
17     extraLessons = (25 - age) * 2
18     print ("You need", baseLessons + extraLessons, "lessons.")
19 else:
20     print ("You need", baseLessons, "lessons.")
21
22 print ("Goodbye")

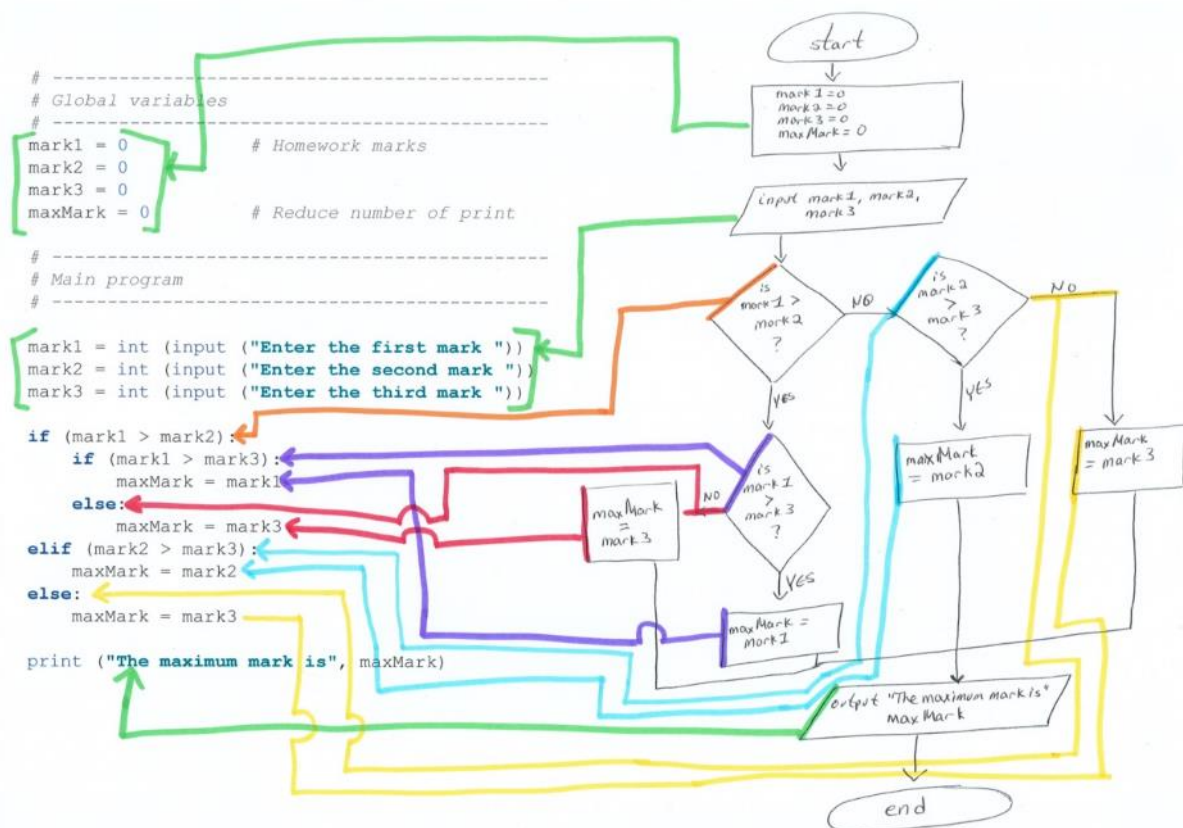
```

## Activity 16 – Page 43

The activity illustrates the connections between the flowcharts and the program code.

Use the flowchart from Figure 1.4.7 and the code from the maximum homework worked example above.

Match up the flowchart symbols to the corresponding section of program code.



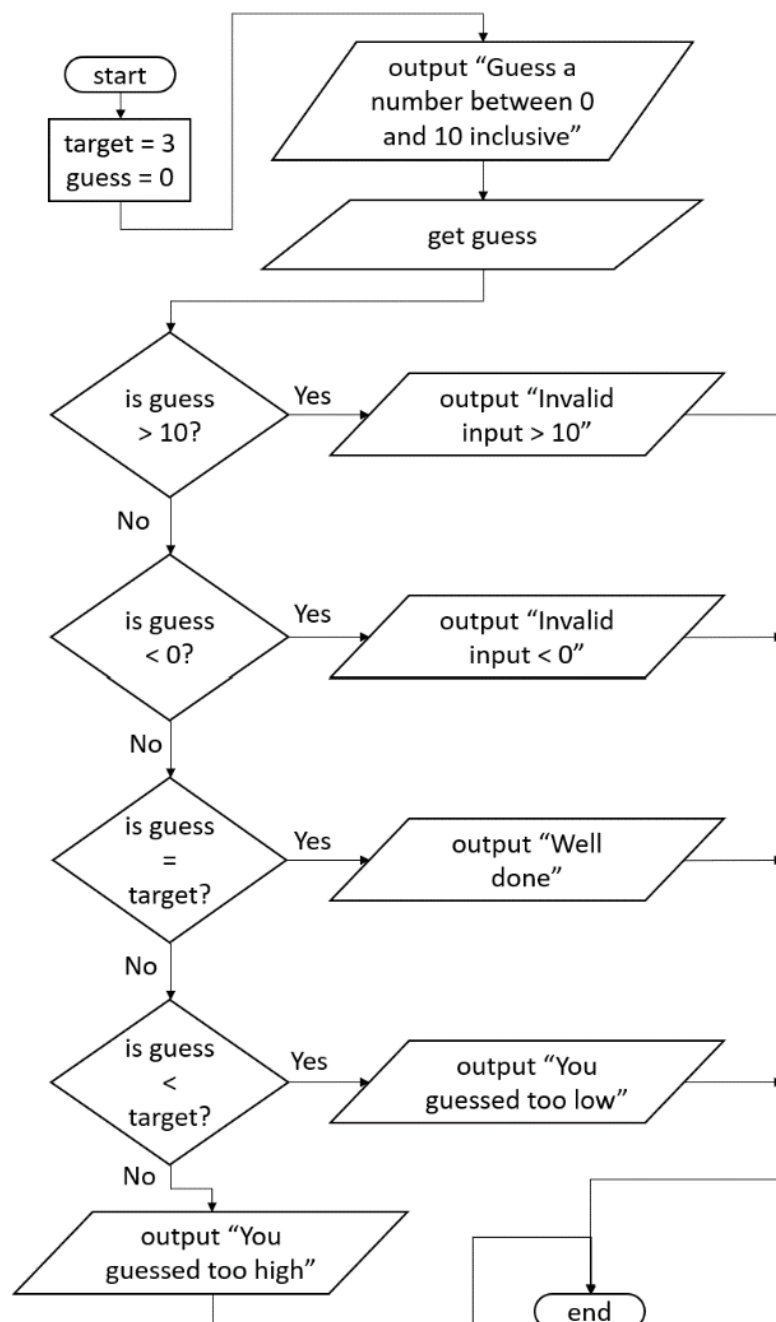
## 1.4: Selection and relational operators

### Activity 17 – Page 43

A learner is creating a guessing game. A player has to enter a number between 1 and 10, inclusive. If the number is too high, they are informed of an error. If the number is too low, they are informed of an error. If the guess is within the range 1 to 10, they are told whether or not they have guessed too high, too low or the correct number. (Assume that the correct number is 3.)

Create an algorithm to solve this problem. Express it as a flowchart and translate it into programming code.

*Solution flowchart*



## 1.4: Selection and relational operators

### Solution code

```

1  # -----
2  # Global variables
3  # -----
4  target = 3          # The number to guess
5  guess = 0          # The user guesses this
6
7  # -----
8  # Main program
9  # -----
10 guess = int (input ("Guess a number between 0 and 10, inclusive "))
11
12 if (guess > 10):
13     print ("Invalid input, number greater than 10")
14 elif (guess < 0):
15     print ("Invalid input, number less than 0")
16 elif (guess == target):
17     print ("Well done!")
18 elif (guess < target):
19     print ("You guessed too low")
20 else:
21     print ("You guessed too high")
22

```

### PRIMM activity - Page 43

Here is the code for a program that involves fruit and boxes. Here are the names of some fruits that will make good test data for the program. In a later topic, you'll learn how to select good test data, but for now, you can use these fruit names.

raspberry	huckleberry	apple
satsuma	jackfruit	coconut
tayberry	kiwifruit	date
fig		

```

1  # -----
2  # Global variables
3  # -----
4  theFruit = ""      # Stores name of a fruit
5
6  # -----
7  # Main program
8  # -----
9  theFruit = input ("Enter the name of a fruit ")
10
11 if (theFruit[0] > 's'):
12     print ("Put " + theFruit + " in box 3")
13 elif (theFruit[0] > 'j'):
14     print ("Put " + theFruit + " in box 2")

```

## 1.4: Selection and relational operators

```

15 else:
16     print ("Put " + theFruit + " in box 1")
17
18 print ("Now, make fruit salad")

```

### Predict

1. What do you think the code will do?

Sorts fruit into boxes based on the first letter of the fruit name

2. What output do you think it will produce?

Tayberry goes to box 3. Kiwifruit and satsuma go to box 2. Date goes to box 1.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes

3. Did the output match your prediction? If not, how did it differ?

The first time I read the program, I put satsuma into box 3. I misread the > symbol as >=.

### Investigate

1. What letters must start the names of fruits that go into box 2?

k, l, m, n, o, p, q, r, and s

2. What box does 'strawberry' go into?

Box 2

3. What would be the effect of changing each '>' symbol to '>='?

Satsuma and strawberry would move from box 2 to box 3. Jackfruit would move from box 1 to box 2.

### Modify

1. Change the program so that 'strawberry' goes into the third box.

```

if (theFruit[0] >= 's'):
    print ("Put " + theFruit + " in box 3")

```

2. Add code to use another box, named "box X", for fruit starting with the letters 'd' to 'j'.

```

elif (theFruit[0] >= 'd'):
    print ("Put " + theFruit + " in box X")

```

3. Would the program work if the order of the test conditions were rearranged, i.e. box 1 is first, box 2 is second, box 3 is third? If so, why? If not, why?

## 1.4: Selection and relational operators

It depends on how the relational operators are written. If the alphabet is processed from the front, then the relational operators must be less than rather than greater than. However, the alphabet must be processed in order. Try it. Mix up the order of the tests and the relational operators to find out for yourself.

### Make

A program is needed to calculate the total price of a purchase, made up of several boxes. Each box costs £2.55. Discounts are applied for purchasing multiple boxes according to this table. Users should be told their level of discount and the total to pay.

1. Express the algorithm as a flowchart.
2. Translate the algorithm to program code.
3. Test and debug your code.

An example console session is shown for you.

```
Enter number of boxes to buy 101
You have a 20% discount. 206.03999999999996
```

Number of boxes	Percentage discount
24 or fewer	No discount
25 to 49	5%
50 to 74	10%
75 to 99	15%

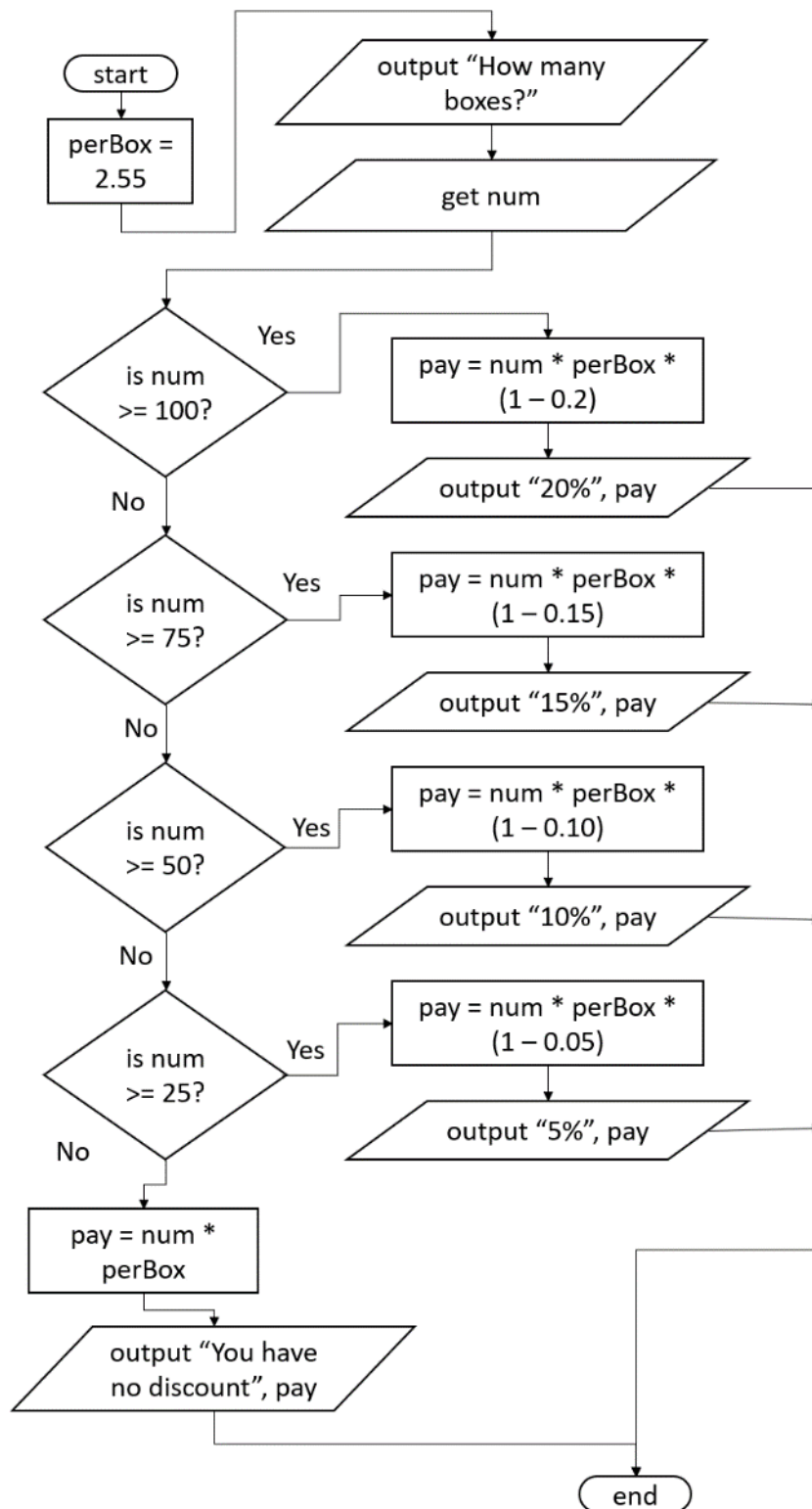
You will learn how to format output in a better way in a later topic. If you want to jump ahead, you can use the `round(x, n)` function. For example, `round(toPay, 2)` would return 206.04.

An example console session is shown for you.

```
Enter number of boxes to buy 101
You have a 20% discount. 206.04
```

## 1.4: Selection and relational operators

Solution flowchart





## 1.4: Selection and relational operators

## Solution code

```

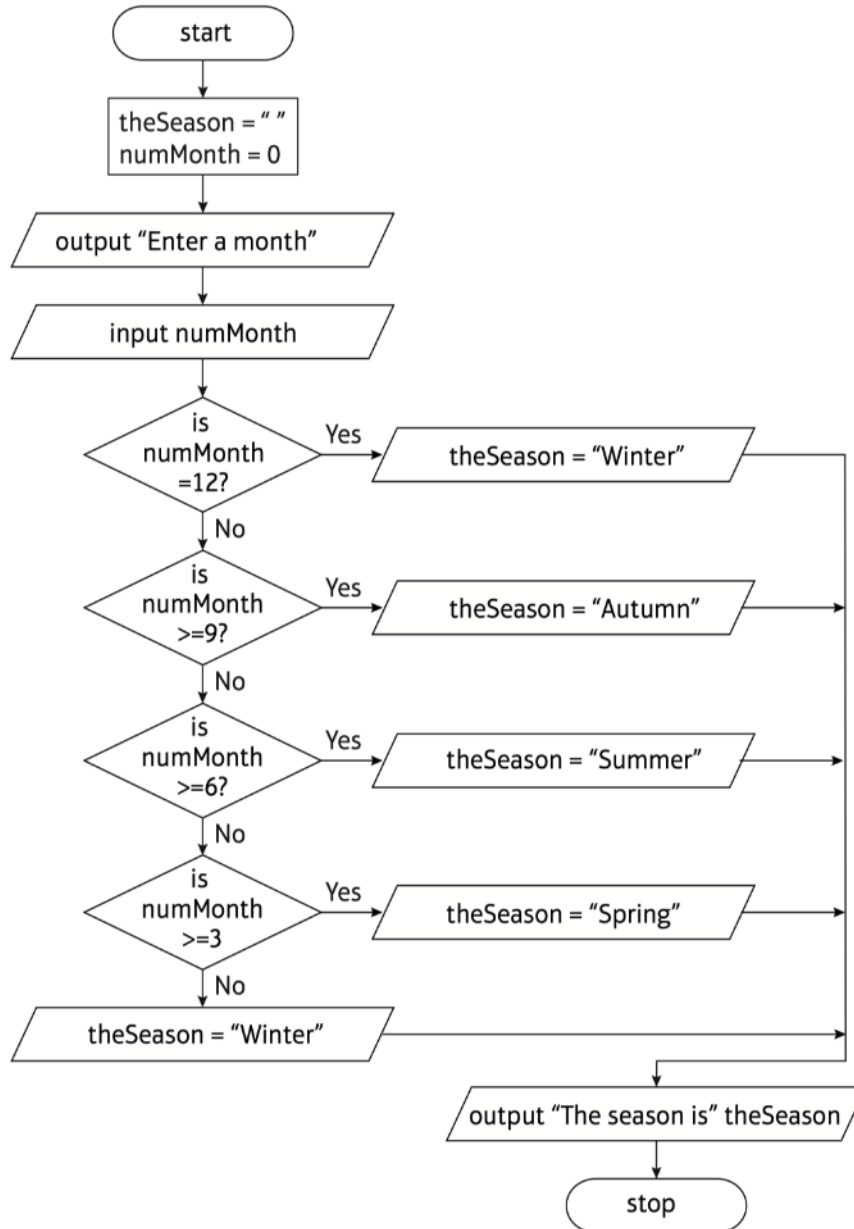
1  # -----
2  # Global variables
3  # -----
4  pricePerBox = 2.55          # Price for each box
5
6  # -----
7  # Main program
8  # -----
9  numBoxes = int (input("Enter number of boxes to buy "))
10
11 if (numBoxes >= 100):
12     toPay = numBoxes * pricePerBox * (1 - 0.2)
13     print ("You have a 20% discount.", round(toPay, 2))
14 elif (numBoxes >= 75):
15     toPay = numBoxes * pricePerBox * (1 - 0.15)
16     print ("You have a 15% discount.", toPay)
17 elif (numBoxes >= 50):
18     toPay = numBoxes * pricePerBox * (1 - 0.10)
19     print ("You have a 10% discount.", toPay)
20 elif (numBoxes >= 25):
21     toPay = numBoxes * pricePerBox * (1 - 0.05)
22     print ("You have a 5% discount.", toPay)
23 else:
24     toPay = numBoxes * pricePerBox
25     print ("You have no discount.", toPay)

```

## 1.4: Selection and relational operators

## Exam-style questions – Page 45

1. This flowchart is for an algorithm that determines the season based on the month number.



Here is the code layout for a program. Amend the code to implement the flowchart algorithm. (5 marks)

## 1.4: Selection and relational operators

*Provided*

```

1  # -----
2  # Global variables
3  # -----
4  # =====> Put your variables here
5
6  # -----
7  # Main program
8  # -----
9  # =====> Put your code here
10

```

*Solution*

```

1  # -----
2  # Global variables
3  # -----
4  # =====> Put your variables here
5  theSeason = ""
6  numMonth = 0          # Number of the month
7
8  # -----
9  # Main program
10 # -----
11 # =====> Put your code here
12 numMonth = int (input ("Enter a month number "))
13
14 if (numMonth == 12):
15     theSeason = "Winter"
16 elif (numMonth >= 9):
17     theSeason = "Autumn"
18 elif (numMonth >= 6):
19     theSeason = "Summer"
20 elif (numMonth >= 3):
21     theSeason = "Spring"
22 else:
23     theSeason = "Winter"
24 print ("The season is " + theSeason)

```

## 1.4: Selection and relational operators

2. Stock items are directed to different warehouses, based on the stock identification number.

Stock identifiers consist of a letter (A–Z) followed by a number (0–9). Warehouse destinations, for valid stock identifiers, are shown in this table.

Here is the code for the program, but the lines are jumbled up.

Amend the program code by rearranging and indenting the lines to make the program function properly. Do not add any other functionality. Do not change the lines of code.

(7 marks)

*Provided*

```

1  # -----
2  # Global variables
3  # -----
4
5  # -----
6  # Main program
7  # -----
8  print ("This stock goes to " + city)
9  print ("Invalid stock ID")
10 stockID = ""
11 stockID = stockID.upper()
12 city = ""
13 city = "Newcastle"
14 city = "Cardiff"
15 city = "Plymouth"
16 city = "Birmingham"else:
17 elif (stockID[0] <= "R"):
18 if (stockID[0] <= "F"):
19 elif (stockID[0] <= "L"):
20 if (len(stockID) > 2):
21 else:
22 stockID = input("Enter a stock ID ")

```

*Solution*

```

1  # -----
2  # Global variables
3  # -----
4  stockID = ""
5  city = ""
6
7  # -----
8  # Main program
9  # -----
10 stockID = input ("Enter a stock ID ")

```

## 1.4: Selection and relational operators

```
11 stockID = stockID.upper()
12
13 if (len(stockID) > 2):
14     print ("Invalid stock ID")
15 else:
16     if (stockID[0] <= "F"):
17         city = "Birmingham"
18     elif (stockID[0] <= "L"):
19         city = "Cardiff"
20     elif (stockID[0] <= "R"):
21         city = "Plymouth"
22     else:
23         city = "Newcastle"
24
25 print ("This stock goes to " + city)
```

## 1.5: Repetition

### Activity 18 – Page 49

Repetition test condition	Keep the repetition going	Stop the repetition
<code>while (score != 0):</code>	5, 8, 22	0
<code>while (temp &lt; 18.5):</code>	12.3, 0.0, -15.8	18.51, 39.3, 99.22
<code>while (level &gt; 85):</code>	100, 9992	84, 0, -3
<code>while (isValid):</code>	True	False
<code>while (count == 5):</code>	5	-3, 0, 4, 6, 8
<code>while (count != 5):</code>	-2, 0, 4, 6	5
<code>while (isValid == False):</code>	False	True

### Activity 19 – Page 51

- Write a program that asks the user to enter a number between 1 and 6. If the number entered matches the value stored in the variable `diceRoll`, the message “Well done, you guessed correctly” is displayed. Otherwise the user is invited to guess again.

#### Solution

```

1  # -----
2  # Global variables
3  # -----
4  diceRoll = 2          # User is trying to guess this
5  guess = 0            # User types this in
6
7  # -----
8  # Main program
9  # -----
10
11 # Note that setting guess different to diceRoll forces the loop
12 # to execute at least once
13 while (guess != diceRoll):
14     guess = int(input(" Enter a value, 1 to 6 "))
15
16 print ("Well done you guessed correctly")
17 print("Goodbye")

```

## 1.5: Repetition

- Extend the program so that the user is told if they enter a number outside the range 1 to 6.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  diceRoll = 2          # User is trying to guess this
5  guess = 0            # User types this in
6
7  # -----
8  # Main program
9  # -----
10
11 # Note that setting guess different to diceRoll forces the loop
12 # to execute at least once
13 while (guess != diceRoll):
14     guess = int(input(" Enter a value, 1 to 6 "))
15     if (guess < 1):
16         print ("Invalid input")
17     elif (guess > 6):
18         print ("Invalid input")
19     print ("Well done you guessed correctly")
20     print("Goodbye")
21 # -----

```

### Activity 20 – Page 52

Look at each of the examples in the following table. Using your understanding of logical operators and order of precedence, evaluate each expression and state if it evaluates to either True or False.

The first one has been done for you. You might find it helpful to type them into your programming environment to check your answers.

Put each of these conditions inside a print() instruction.

a	b	c	Statement	Evaluates
3	6	8	a == 3 and b == 6	True
5	6	8	a == 3 and b == 6	False
3	6	8	a == 3 or b == 8	True

## 1.5: Repetition

5	6	8	$a > 3$ or $b \neq 8$	True
3	6	8	$a == 5$ or $b < 5$ or $c \geq 8$	True
3	6	8	not ( $a \geq 3$ )	False
3	6	8	not ( $a \leq 3$ and $b == 6$ )	False
3	6	8	not ( $a \leq 3$ and $b \geq 9$ )	True
3	6	8	not $a \leq 3$ and $b == 5$	False
3	6	8	not ( $a \leq 3$ or $b \geq 9$ )	False
3	6	8	not ( $a \leq 3$ ) or $b == 5$	False
3	6	8	$a \leq 3$ and not ( $b == 5$ )	True
3	6	8	(( $a \leq 3$ ) and ( $b - a == 3$ )) and (not ( $c > 10$ ))	True

### Activity 21 - Page 53

The temperature program in the worked example only executes for the one combination of day and temperature which are hard coded into the program. Amend the program to:

- Generalise the program by adding instructions to allow the user to input the day number and the temperature.

#### Solution

```

1  # -----
2  # Global variables
3  # -----
4  # Test with 1, 22; 1, -2.25; 6, -2.25; 7, 35; 5, 25; 5, 25.5
5  day = 5                      # Day of the week
6  temp = 25.5                  # Temperature (real number)
7
8  # -----
9  # Main program
10 # -----
11 if (day >= 1) and (day <=5):
12     # Weekday
13     if (temp < 20.0) or (temp > 25.0):

```



## 1.5: Repetition

```

14         # Adjust temperature
15         print ("Weekday - Need to adjust temperature")
16 elif (day == 6) or (day == 7):
17     # Weekend
18     if (temp < 15.0) or (temp > 30.0):
19         # Adjust temperature
20         print ("Weekend - Need to adjust temperature")

```

2. Add a repetition so that the program goes around the loop until the user enters a day of 0.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  # Test with 1, 22; 1, -2.25; 6, -2.25; 7, 35; 5, 25; 5, 25.5
5  day = 5                                # Day of the week
6  temp = 25.5                            # Temperature (real number)
7
8  # -----
9  # Main program
10 # -----
11 day = int (input ("Enter a day of the week, 1 to 7 "))
12
13 while (day != 0):                        # Stop when user says
14     temp = float(input("Enter a temperature "))
15     if (day >= 1) and (day <=5):
16         # Weekday
17         if (temp < 20.0) or (temp > 25.0):
18             # Adjust temperature
19             print ("Weekday - Need to adjust temperature")
20     elif (day == 6) or (day == 7):
21         # Weekend
22         if (temp < 15.0) or (temp > 30.0):
23             # Adjust temperature
24             print ("Weekend - Need to adjust temperature")
25     day = int(input("Enter a day of the week, 1 to 7 "))

```

### Activity 22 - Page 55

In a previous activity, you created an algorithm for a guessing game. This is a different type of guessing game and it is a bit more difficult. The program generates a random number between 1 and 20. The user only has three guesses to find it.

The program is required to:

- generate a random integer between 1 and 20, inclusive

## 1.5: Repetition

- ask the user to guess the number
- allow the user three attempts
- display a message if the attempt is correct
- display a message if the attempt is incorrect and inform the player if their attempt is too high or too low
- display a message to the player after three incorrect attempts informing them of the correct number.

### Solution

```

1  # -----
2  # Import Libraries
3  # -----
4  import random                # Library for random numbers
5
6  # -----
7  # Global variables
8  # -----
9  theTarget = 0                # The number to guess
10 maxGuesses = 3               # Allow three attempts
11 countGuess = 0               # Which number guess this is
12 userGuess = 0                # The user's guess, typed in
13
14 # -----
15 # Main program
16 # -----
17 theTarget = random.randint(1,20) # Generate the target number
18 print ("Sneaky peek just for debugging: ", theTarget)
19
20 while (userGuess != theTarget) and (countGuess < maxGuesses):
21     # Ask the user to guess the number
22     userGuess = int(input("Enter a number between 1 and 20 "))
23
24     if (userGuess == theTarget):    # Right answer
25         print ("Well done, the number was " + str (theTarget))
26     elif (userGuess < theTarget):    # Guess too Low
27         print ("Your guess was too low")
28         countGuess = countGuess + 1
29         if (countGuess == maxGuesses): # Ran out of guesses
30             print("The correct number is " + str(theTarget))
31         else:
32             print ("You have " + str(maxGuesses - countGuess) + " guesses left")
33     else:                            # Must be too high
34         print ("Your guess was too high")
35         countGuess = countGuess + 1
36         if (countGuess == maxGuesses): # Ran out of guesses
37             print("The correct number is " + str(theTarget))

```

## 1.5: Repetition

```

38         else:
39             print ("You have " + str(maxGuesses - countGuess) + " guesses left")
40
41     print ("Goodbye")

```

### Activity 23 - Page 57

This activity is yet another variation on the guess the number game. This time, you're going to choose the target number and the number of possible guesses. Then, the computer program will attempt to guess your number.

A console session is shown here. From it you can identify the rules of the game. Do not worry if the computer guesses the same number twice.

Console session:

```

Enter a number between 1 and 20 10
How many guesses will you give the computer? 20
Guess number: 1 was 8
Guess number: 2 was 17
Guess number: 3 was 5
Guess number: 4 was 5
Guess number: 5 was 2
Guess number: 6 was 19
Guess number: 7 was 4
Guess number: 8 was 16
Computer wins by guessing 10 on turn number 9
Goodbye

```

### Solution

```

1  # -----
2  # Import Libraries
3  # -----
4  import random                # For random integer generation
5
6  # -----
7  # Global variables
8  # -----
9  theTarget = 0                # The number to guess
10 maxNumGuesses = 10           # Maximum number of guesses
11 numGuess = 0                 # Current number of guess
12 randomGuess = 0              # The computer's guess
13 correct = False              # Computer hasn't guess correctly
14
15 # -----
16 # Main program

```

## 1.5: Repetition

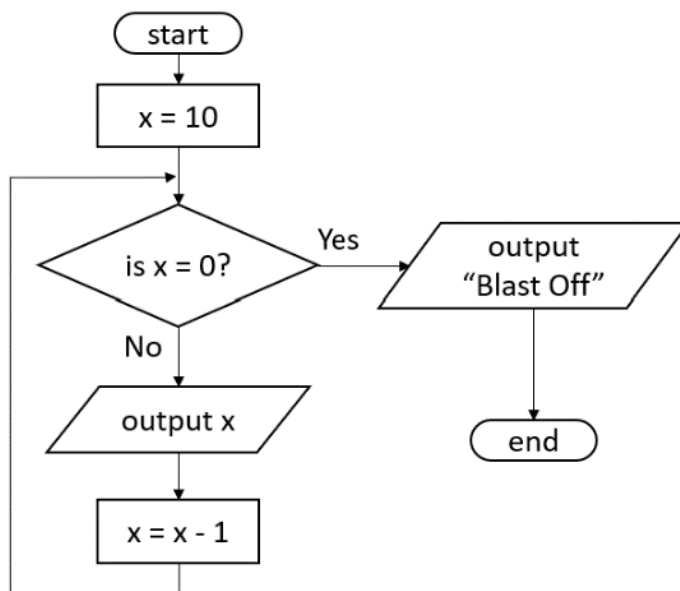
```

17 # -----
18 theTarget = int (input ("Enter a number between 1 and 20 "))
19 maxNumGuesses = int (input ("How many guesses will you give the computer? "))
20
21 # while (correct == False) and (numGuess < maxNumGuesses):
22 while (not correct) and (numGuess < maxNumGuesses):
23     randomGuess = random.randint(1, 20)
24     if (randomGuess == theTarget):
25         print ("Computer wins by guessing " + str(randomGuess) +
26               " on turn number " + str(numGuess+1))
27         correct = True
28     else:
29         print ("Guess number: " + str(numGuess+1) + " was " +
30               str(randomGuess))
31         numGuess = numGuess + 1
32
33 print ("Goodbye")

```

### Activity 24 - Page 60

'Blast off' is a program to simulate a rocket countdown. Translate the blast off algorithm, shown in the flowchart, to program code. Remember to use appropriate techniques to make it easier to read.



### Solution

```

1 # -----
2 # Global variables
3 # -----
4 count = 10

```

## 1.5: Repetition

```

5
6 # -----
7 # Main program
8 # -----
9 while (count > 0):
10     print (count)
11     count = count - 1
12 print ("Blast Off!!")

```

### Activity 25 – Page 60

Write a program that simulates a simple calculator. It must:

1. allow the user to choose from these options: addition, subtraction, division and multiplication
2. prompt the user to input two numbers
3. perform the calculation and displays the result
4. offer the user the option of performing another calculation.

#### Solution

```

1 # -----
2 # Global variables
3 # -----
4 num1 = 0.0
5 num2 = 0.0                                # Real numbers
6 userChoice = ""                           # A, S, D, M
7 goAgain = True                            # Force loop first time
8
9 # -----
10 # Main program
11 # -----
12 while (goAgain):
13     # Get user choice
14     userChoice = input ("Choose (A)ddition, (S)ubtraction, (D)ivision,
15 (M)ultiplication ")
16
17     # Do the calculation
18     if (userChoice == "A"):
19         num1 = float (input ("Enter the first number "))
20         num2 = float (input ("Enter the second number "))
21         print (num1 + num2)
22     elif (userChoice == "S"):
23         num1 = float (input ("Enter the first number "))
24         num2 = float (input ("Enter the second number "))
25         print (num1 - num2)
26     elif (userChoice == "D"):

```

## 1.5: Repetition

```

26         num1 = float (input ("Enter the first number "))
27         num2 = float (input ("Enter the second number "))
28         print (num1 / num2)
29     elif (userChoice == "M"):
30         num1 = float (input ("Enter the first number "))
31         num2 = float (input ("Enter the second number "))
32         print (num1 * num2)
33
34     # Find out if the user wants to do another
35     userChoice = input ("Do you want to calculate more? Y/N ")
36     if (userChoice == "Y"):
37         goAgain = True
38     else:
39         goAgain = False
40
41     print ("Goodbye")

```

### PRIMM activity – Page 61

```

1  r = 5
2  c = 0
3  out = ""
4  coord = ""
5  while (r >= 0):
6      c = 0
7      out = ""
8      while (c < 4):
9          coord = "(" + str(r) + "," + str(c) + ")"
10         out = out + " " + coord
11         c = c + 1
12     print (out)
13     r = r - 1
14     print ("Goodbye")

```

#### Predict

Displays quote marks with two values inside them, separated by a comma (line 9).

#### Run

Yes, but it printed them in a block of four across and six down. There are 24 things called 'coord', short for coordinates.

#### Investigate

1. What happens if the variable r is initialised to 1?

Only two rows are printed, by four items across.

## 1.5: Repetition

2. What happens if the variable *c* is initialised to 1?

There doesn't appear to be any change in the output. This is because the value of *c* controls the inside loop and is reset to zero each time before the inside loop starts.

3. What happens if the conditions on the outside while loop are changed from  $\geq$  to  $>$ ?

One less row is printed. There are five rows instead of six.

4. Reload the original code and run it.
5. Have you seen this type of coordinate system in other subjects?

Yes, this is the Cartesian coordinate system, as used in mathematics and the sciences.

### Modify

1. Amend the code so that it is more readable.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  row = 5
5  column = 0                                # Origin 0
6  outString = ""                            # Empty to start
7  coordinates = ""                          # Cartesian coordinates
8
9  # -----
10 # Main program
11 # -----
12
13 while (row >= 0):
14     column = 0
15     outString = ""                        # Next Line
16     while (column < 4):
17         coordinates = "(" + str(row) + "," + str(column) + ")"
18         outString = outString + " " + coordinates
19         column = column + 1
20     print (outString)
21     row = row - 1
22
23 print ("Goodbye")

```

### Make

A program is needed that will output part of the times table for integers. The program must:

- Ask the user for a number between 1 and 10.
- Print the times table for that number, from 1 to that number.
- Let the user choose to exit by entering a 0.

An example console session is shown for you.

Enter a number between 1 and 10, 0 to quit 5

## 1.5: Repetition

1 x 5 = 5  
 2 x 5 = 10  
 3 x 5 = 15  
 4 x 5 = 20  
 5 x 5 = 25  
 Enter a number between 1 and 10, 0 to quit 0  
 Goodbye

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  maxNum = 0
5  count = 1
6
7  # -----
8  # Main program
9  # -----
10 maxNum = int (input ("Enter a number between 1 and 10, 0 to quit "))
11
12 while (maxNum != 0):
13     count = 1
14     while (count <= maxNum):
15         print (str(count) + " x " + str(maxNum) + " = " + str(count * maxNum))
16         count = count + 1
17     maxNum = int (input ("Enter a number between 1 and 10, 0 to quit "))
18
19 print ("Goodbye")
  
```



## 1.5: Repetition

### Exam-style questions – Page 62

1. XtraSave is a chain of supermarkets that has recently installed self-checkouts in all its stores.

Customers are issued with one 10p voucher for each £10 they spend and one 5p voucher if they spend £5.

The code for the self-checkout system is shown here, but it is difficult to read and understand.

Amend the program using brackets, comments, descriptive identifiers, indentation and whitespace to make the program easier to read. (5 marks)

```

1  s = 0
2  sf = 0.0
3  sf = float (input ("Enter the amount spent "))
4  s = int (sf)
5  print ("Nearest whole pound Sterling:", s)
6  while (s >= 5):
7      if (s >= 10):
8          print ("Printing 10p voucher")
9          s = s - 10
10     elif (s >= 5):
11         print ("Printing 5p voucher")
12         s = s - 5
13 print ("Thank you for shopping with us")

```

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  amountSpent = 0                # Customer spend in integer
5  amountSpentFloat = 0.0        # Possible to spend decimals
6  # -----
7  # Main program
8  # -----
9
10 amountSpentFloat = float (input ("Enter the amount spent "))
11 amountSpent = int (amountSpentFloat)
12 print ("Nearest whole pound Sterling:", amountSpent)
13
14 while (amountSpent >= 5):
15     if (amountSpent >= 10):
16         print ("Printing 10p voucher")
17         amountSpent = amountSpent - 10
18     elif (amountSpent >= 5):
19         print ("Printing 5p voucher")
20         amountSpent = amountSpent - 5
21 print ("Thank you for shopping with us")

```

## 1.5: Repetition

2. Amend the code in each example to correct the logic errors. (3 marks)

```

1  # -----
2  # Global variables
3  # -----
4  index = int()
5  index = 0
6
7  # -----
8  # Main program
9  # -----
10 # Example 1
11 index = 1
12 while (index < 10):
13     print ("Index = ", index)
14
15 # Example 2
16 index = 1
17 while (index < 10):
18     print ("Index = ", index)
19     index = index - 1
20
21 # Example 3
22 index = 1
23 while (index < 1):
24     print ("Index = ", index)
25     index = index + 1

```

**Solution**

```

1  # -----
2  # Global variables
3  # -----
4  index = int()
5  index = 0
6
7  # -----
8  # Main program
9  # -----
10 # Example 1
11 index = 1
12 while (index < 10):
13     print ("Index = ", index)
14     index = index + 1      # Added this line
15
16 # Example 2
17 index = 1
18 while (index < 10):
19     print ("Index = ", index)

```

## 1.5: Repetition

```

20     index = index + 1           # Changed - to +
21
22 # Example 3
23 index = 1
24 while (index < 10):           # Changed test condition to < 10
25     print ("Index = ", index)
26     index = index + 1

```

- Write a program to generate 10 random numbers between 0 and 100. For each number, report the number and which quartile it is in. Ask the user if they'd like to generate another set of numbers. Use 'Y' to keep looping. Remember to use techniques to make your code readable.

Quartiles are identified in this table.

Range of numbers	Quartile
0 to 24	1
25 – 49	2
50 – 74	3
75 - 100	4

### Solution

```

1  # -----
2  # Import libraries
3  # -----
4  import random
5
6  # -----
7  # Global variables
8  # -----
9  count = 0           # Numbers to generate
10 randomNumber = 0    # Random number
11 userChoice = "Y"    # To keep going
12 # -----
13 # Main program
14 # -----
15
16 while (userChoice == "Y"):
17     # Generate 10 numbers
18     count = 0        # Reset counter
19     while (count < 10):
20         randomNumber = random.randint (0, 100)
21
22         # Check highest first
23         if (randomNumber > 75):

```

## 1.5: Repetition

```
24         print (str(randomNumber) + " is in quartile 4")
25     elif (randomNumber > 50):
26         print (str(randomNumber) + " is in quartile 3")
27     elif (randomNumber > 25):
28         print (str(randomNumber) + " is in quartile 2")
29     else:
30         print (str(randomNumber) + " is in quartile 1")
31
32     count = count + 1          # Next one
33
34     # Does user want another go?
35     userChoice = input ("Do you want another set? ")
36     userChoice = userChoice.upper()
37
38     print ("Goodbye")
```

## 1.6: One-dimensional data structures

## Activity 26 - Page 66

Complete the table to identify and give examples of arrays and records. The first one has been done for you.

Example	Data structure	Data types
<code>myFavourites = ["Blue", "Red", "Purple"]</code>	Array	All strings
<code>scores = [85, 43, 65, 52]</code>	Array	All integers
<code>aCourse = ["G5X3", "French", 1, 34, 17.38, True]</code>	Record	String, string, integer, integer, real, Boolean
<code>mySubjects = ["French", "Maths", "English", "Drama", "Art"]</code>	Array	All strings
<code>theReadings = [88.34, 77.23, 66.12, 33.90, 22.56]</code>	Array	All real
<code>magazine = ["Davies, R.", 2006, "Computing 4 Beginners", 15, 4]</code>	Record	String, integer, string, integer, integer
<code>refreshments = [True, False, False, True, True, False]</code>	Array	All Boolean

## Activity 27 – Page 68

Here is a set of data structures.

`magazine = ["Davies, R.", 2006, "Computing 4 Beginners", 15, 4]`

`theReadings = [88.34, 77.23, 66.12, 33.89, 22.56]`

`scores = [85, 43, 65, 52]`

`aCourse = ["G5X3", "French", 1, 34, 17.38, True]`

Complete the table using these data structures to find the values and expressions for the blank cells.

Expression	Result
<code>scores[2]</code>	65
<code>aCourse[1]</code>	French

## 1.6: One-dimensional data structures

<code>len(theReadings)</code>	5
<code>not aCourse[5]</code>	False
<code>theReadings[0] + aCourse[4]</code>	105.72
<code>aCourse[1] + magazine[2]</code>	FrenchComputing 4 Beginners
<code>len(scores) + aCourse[3]</code>	38

## Activity 28 – Page 68

Write a program to meet the following requirements. Hint: copy these requirements into your program as comments and write the lines of program code under each of them.

- Declare and initialise an array of five pet names.
- Declare and initialise a list of the prices of six books.
- Declare and initialise a record for the number 98 bus, carrying 50 passengers, going to Oxford, leaving at 8:32 a.m.
  - How will you store 08:32? Is it a number or something else?
- Print the length of each data structure.
- Print the first item in each data structure using indexing.
- Calculate the index of the last item in each data structure using `len()`.
- Print the last item using this calculation.
- Change the price of the third book.
- Print the price of the fourth book using indexing.

## Solution

```

1  -----
2  # Global variables
3  # -----
4  pets = ["Reba", "Spot", "Fred", "Rocky", "Tiger"]
5  bookPrices = [10.23, 5.80, 8.00, 5.60, 6.30, 7.12]
6  busRecord = [98, 50, "Oxford", "08:32"]
7
8  # -----
9  # Main program
10 # -----
11 print ("Length of all data structures")
12 print ("Length pets:", len(pets))
13 print ("Length bookPrices:", len(bookPrices))
14 print ("Length busRecord:", len(busRecord))
15
16 print ("First item in all data structures")
17 print (pets[0])
18 print (bookPrices[0])
19 print (busRecord[0])

```

## 1.6: One-dimensional data structures

```

20
21 print ("Last item in all data structures, using len()")
22 print (pets[len(pets)-1])
23 print (bookPrices[len(bookPrices)-1])
24 print (busRecord[len(busRecord)-1])
25
26 print ("Changing the price of first book and reprinting")
27 print ("Price of 3rd book is", bookPrices[2])
28 bookPrices[2] = 22.22
29 print ("Price of 3rd book is", bookPrices[2])

```

### Activity 29 – Page 70

Write a program to meet the following requirements.

Hint: copy these requirements into your program as comments and write the lines of program code under each of them.

- Declare and initialise an empty array called distanceCities.
- Print the length of distanceCities.
- Print the type of distanceCities. Hint: use `type()`.
- Print the contents of distanceCities.
- Add 1528.13, 721.55, 810.38 to distanceCities.
- Print the length of distanceCities.
- Print the contents of distanceCities.
- Insert 333.33 after 721.55.
- Print the contents of distanceCities.
- Delete 721.55.
- Print the contents of distanceCities.
- Calculate the index of the last item in each data structure using `len()`.
- Print the last item using this calculation.
- Change the first item to 1111.11.
- Print the contents of distanceCities.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  distanceCities = []           # Empty List
5  lastIndex = 0                # Calculated Later
6
7  # -----
8  # Main program
9  # -----
10
11 # Print contents and Length
12 print (len(distanceCities))
13 print (type(distanceCities))

```

## 1.6: One-dimensional data structures

```

14 print (distanceCities)
15 distanceCities.append(1528.13)
16 distanceCities.append(721.55)
17 distanceCities.append(810.38)
18 print (len(distanceCities))
19 print (distanceCities)
20
21 # Insert and delete
22 distanceCities.insert(2, 333.33)
23 print (distanceCities)
24 del distanceCities[1]
25 print (distanceCities)
26
27 # Last and first item
28 lastIndex = len(distanceCities) - 1
29 print (distanceCities[lastIndex])
30 distanceCities[0] = 1111.11
31 print (distanceCities)

```

### Activity 30 – page 74

A data structure is provided here which contains a list of integers.

```
theNumbers = [23, 75, 15, 88, 21, 99]
```

A program is needed to find the sum and average of all the integers.

Report the sum and average as whole integers.

1. Write the program using a `for...in range()` loop.
2. Write the program using a `for...in` data structure loop.

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  theNumbers = [23, 75, 15, 88, 21, 99]
5  theSum = 0                                # Adding up
6  theAverage = 0
7  index = 0                                # Use in for...range()
8  num = 0                                   # Use in for each
9
10 # -----
11 # Main program
12 # -----
13
14 for index in range (0, len(theNumbers)):
15     theSum = theSum + theNumbers[index]
16 theAverage = theSum / len(theNumbers)
17 theAverage = int (round (theAverage, 0))

```



## 1.6: One-dimensional data structures

```

18 print ("The Sum = " + str(theSum) +
19       " and the Average = " + str(theAverage))
20
21 theSum = 0
22 for num in theNumbers:
23     theSum = theSum + num
24 theAverage = theSum / len(theNumbers)
25 theAverage = int (round (theAverage, 0))
26 print ("The Sum = " + str(theSum) +
27       " and the Average = " + str(theAverage))

```

### PRIMM activity – Page 77

A school is keeping track of how many books the students have read over the summer holidays. The two lists are parallel, i.e. Alex has read 12 books and Carol has read 5 books.

```

1  # -----
2  # Global variables
3  # -----
4  allStudents = ["Alex", "Eloise", "Bryn", "Lois", "Carol", "Kevin"]
5  booksRead = [12, 25, 18, 7, 5, 14]
6
7  theStudent = ""                # One student
8  numBooks = 0                   # Number of books read
9  index = 0
10 length = 0                     # Length of list
11 layout = ""                    # Formatting
12
13 # -----
14 # Main program
15 # -----
16 length = len (allStudents)      # To aid readability
17
18 # Loop through both lists together using the same index
19 for index in range (0, length):
20     theStudent = allStudents[index]
21     numBooks = booksRead[index]
22
23     # Output the details
24     layout = "{:<5}{:>2d}"
25     print (layout.format(theStudent, numBooks))

```

### Predict

1. What do you think the code will do?

Prints the name of the student followed by the number of books read by that student.

2. What output do you think it will produce?

The student name and a number, in a columnar format.

## 1.6: One-dimensional data structures

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes, it printed students name and a number.

3. Did the output match your prediction? If not, how did it differ?

Not exactly. Eloise and Keven have the number pushed against their names. There should be at least one space between the name and the number.

### Investigate

1. Why must both allStudents and booksRead have the same number of items?

Because if students was too short, the books would be too long. It wouldn't be obvious which number of books went with each student.

2. Would the outcome be different, if len(allStudents) was put inside the parameters for the 'for' loop, rather than into a separate variable?

No, the output would still be the same because the upper bound on the range would be calculated as the length of allStudents.

3. Is the output suitable for the audience and fit for purpose?

No, because Eloise and Kevin need a space before the number.

4. Look at this line:

```
layout = "{:<5} {:>2d}"
```

5. Given that the symbol '^' centres, what will the symbols '<' and '>' do?

Control alignment. The symbol < means left align; the symbol > means right align.

6. What type value will go into the second positional argument?

An integer, which is the number of books.

7. What will the '2d' do in the second field?

It says that the value will be an integer to be formatted into two columns.

### Modify

Correct the output, adding headers and spacing to make it more readable

## 1.6: One-dimensional data structures

Name	Num
-----	
Alex	12
Eloise	25
Bryn	18
Lois	7
Carol	5
Kevin	14

*Solution*

```

1  # -----
2  # Global variables
3  # -----
4  allStudents = ["Alex", "Eloise", "Bryn", "Lois", "Carol", "Kevin"]
5  booksRead = [12, 25, 18, 7, 5, 14]
6
7  theStudent = ""                # One student
8  numBooks = 0                   # Number of books read
9  index = 0
10 length = 0                     # Length of list
11 layout = ""                   # Formatting
12
13 # -----
14 # Main program
15 # -----
16 length = len (allStudents)     # To aid readability
17
18 # Print out some headers and a border
19 layout = "{:<6}  {:<3}"
20 print (layout.format ("Name", "Num"))
21 print ("-"*15)
22
23 # Loop through both lists together using the same index
24 for index in range (0, length):
25     theStudent = allStudents[index]
26     numBooks = booksRead[index]
27
28     # Output the details
29     layout = "{:<6}  {:>3d}"
30     print (layout.format(theStudent, numBooks))

```

**Make**

A list of flower names is provided for you. Currently, these are in all lower case.

## 1.6: One-dimensional data structures

```
allFlowers = ["myrtle", "violet", "fuchsia", "buttercup", "daisy",
              "rose", "aster", "orchid", "tulip", "camelia"]
```

A table should be output showing the data in the list and a 'Y' or 'N' to indicate if the flower is in stock. Only half the stock has arrived, so any flower starting with the letters L-Z are not available.

### Solution

```
1  # -----
2  # Global variables
3  # -----
4  allFlowers = ["myrtle", "violet", "fuchsia", "buttercup", "daisy",
5               "rose", "aster", "orchid", "tulip", "camelia"]
6
7  layout = ""                # For formatting
8  inStock = "N"              # For last column
9  # -----
10 # Main program
11 # -----
12
13 layout = "{:9}  {:^9}"
14 print (layout.format("Name", "Available"))
15 print ("-"*20)
16 for flower in allFlowers:
17     if (flower[0] >= "l"):
18         inStock = "N"
19     else:
20         inStock = "Y"
21     print (layout.format(flower, inStock))
```

## 1.6: One-dimensional data structures

## Exam-style questions – Page 78

Here is the output of a program that manipulates weather data. The output is suitable for the audience and fit for purpose.

Weekday	Pressure	Max Temp
Monday	1012.80	13.0
Tuesday	1023.20	12.0
Wednesday	1038.00	13.0
Thursday	1031.10	12.0
Friday	1019.80	9.0
Saturday	1015.70	10.0
Sunday	994.90	11.0
Average	1019.36	11.4

Here is the partially complete code for the program that produced this output.

```

1  # -----
2  # Global variables
3  # -----
4  dayArray = ["Monday", "Tuesday", "Wednesday", "Thursday",
5             "Friday", "Saturday", "Sunday"]
6  rowLabels = ["Pressure", "Max Temp"]
7  pressureArray = [1012.8, 1023.2, 1038.0, 1031.1,
8                  1019.8, 1015.7, 994.9]
9  maxTemperatures = [13, 12, 13, 12, 9, 10, 11]
10
11 index = 0
12 total = -1.0
13 avgPressure = 0.0
14 avgMaxTemp = 0.0
15
16 # -----
17 # Main program
18 # -----
19
20 # Calculate average barometric pressure
21 # =====> Add a FOR EACH loop to calculate the average of all the
22 #           items in pressureArray
23
24 # Calculate average maximum temperature
25 # =====> Add a FOR IN RANGE() loop to calculate the average of
26 #           all the items in maxTemperatures
27
28 # Display titles for columnar output

```

## 1.6: One-dimensional data structures

```

29 # =====> Display column headings for
30 #     weekday (left, 10 wide),
31 #     pressure (centred, 10 wide),
32 #     two blank spaces,
33 #     maximum temperature (centred, 12 wide).
34
35 # =====> Display a separator line (34 wide)
36
37 # Display data values in columnar output
38 # =====> Set up a format string for
39 #     day of the week (left, 10 wide),
40 #     pressure (centred, 10 wide, 2 decimals, float) and
41 #     max temperature (centred, 12 wide, 1 decimal, float)
42
43 # =====> Add a loop to print data from all three arrays
44
45 # Display average values as footer
46 # =====> Display a separator
47
48 # =====> Display average values

```

Amend the code to produce the required output. Follow the instructions given for each block of code, shown in the lines with =====>.

**Hint:** In this question, work from top to bottom, making sure each block of code works as intended before moving onto the next. (15 marks)

Award marks for:

- adding FOR EACH loop for pressureArray (1)
- adding each pressure inside loop (1)
- calculating average pressure using len(pressureArray) (1)
- adding FOR IN RANGE loop for maxTemperatures (1)
- adding each temperature inside loop (1)
- calculating average temperature using len(maxTemperatures) (1)
- only global variables provided (pressure, index, total, avgPressure, avgMaxTemp) are used (1)
- translation of description of column headings to layout string (1)
- display of separator line long enough to cover all columns (1)
- translation of description of column contents to layout string (1)
- using a loop to print every row in the pressureArray (1)
- using a FOR EACH loop to print every row in the pressureArray (1)
- display a separator line for the footer (1)
- print the averages using same layout (1)
- fully functional (1)

## 1.6: One-dimensional data structures

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  dayArray = ["Monday", "Tuesday", "Wednesday", "Thursday",
5             "Friday", "Saturday", "Sunday"]
6  rowLabels = ["Pressure", "Max Temp"]
7  pressureArray = [1012.8, 1023.2, 1038.0, 1031.1,
8                  1019.8, 1015.7, 994.9]
9  maxTemperatures = [13, 12, 13, 12, 9, 10, 11]
10
11 index = 0
12 total = -1.0
13 avgPressure = 0.0
14 avgMaxTemp = 0.0
15
16 # -----
17 # Main program
18 # -----
19
20 # Calculate average barometric pressure
21 # =====> Add a FOR EACH loop to calculate the average of all the
22 #           items in pressureArray
23 total = 0.0
24 for pressure in pressureArray:
25     total = total + pressure
26 avgPressure = total / len(pressureArray)
27
28 # Calculate average maximum temperature
29 # =====> Add a FOR IN RANGE() loop to calculate the average of
30 #           all the items in maxTemperatures
31 total = 0.0
32 for index in range(0, len(maxTemperatures)):
33     total = total + maxTemperatures[index]
34 avgMaxTemp = total / len(maxTemperatures)
35
36 # Display titles for columnar output
37 # =====> Display column headings for
38 #           weekday (left, 10 wide),
39 #           pressure (centred, 10 wide),
40 #           two blank spaces,
41 #           maximum temperature (centred, 12 wide).
42 layout = "{:10} {:^10} {:^12}"
43 print(layout.format("Weekday", "Pressure", "Max Temp"))
44 # =====> Display a separator line (24 wide)
45 print("-"*34)

```

## 1.6: One-dimensional data structures

```

46
47 # Display data values in columnar output
48 # =====> Set up a format string for
49 #     day of the week (left, 10 wide),
50 #     pressure (centred, 10 wide, 2 decimals, float) and
51 #     max temperature (centred, 12 wide, 1 decimal, float)
52 layout = "{:<10} {:^10.2f} {:^12.1f}"
53 # =====> Add a loop to print data from all three arrays
54 for index in range (0, len(pressureArray)):
55     print (layout.format(dayArray[index],
56                           pressureArray[index],
57                           maxTemperatures[index]))
58
59 # Display average values as footer
60 # =====> Display a separator
61 print ("-"*34)
62 # =====> Display average values
63 print (layout.format("Average", avgPressure, avgMaxTemp))

```



## 1.7: Subprograms

### Activity 31 – Page 83

A grid cell reference consists of the letters X, Y or Z, followed immediately by a number 1, 2 or 3. For example, X1, Y2, Z3 are all valid grid references. This program generates valid grid cell references.

```

1  # -----
2  # Import Libraries
3  # -----
4  import random                # Needed library
5
6  # -----
7  # Global variables
8  # -----
9  cellRef = ""                 # String value
10
11 # -----
12 # Subprograms
13 # -----
14 # ==> Write the function code here
15
16 # -----
17 # Main program
18 # -----
19 cellRef = genCellRef()        # Call to function

```

It is missing the code for the function genCellRef().

1. Write the function to complete the program. The function should not print.
2. Make a backup of the program, because it will be used in Step 5.
3. Amend the program to use a repetition (WHILE loop) to print 10 cell references.
4. Amend your program to remove the individual print statements, add each reference to a 1-dimensional data structure, and print the entire data structure in one go.
5. Reload the backup copy of the program from Step 2.
6. Change genCellRef() from a function to a procedure, that prints out each cell reference generated. The main program should not print.

*Solution step 1:*

```

1  # -----
2  # Import Libraries
3  # -----
4  import random                # Needed library
5
6  # -----
7  # Global variables
8  # -----
9  cellRef = ""                 # String value
10

```

## 1.7: Subprograms

```

11 # -----
12 # Subprograms
13 # -----
14 # ==> Write the function code here
15 def genCellRef():
16     theRef = ""           # The value returned
17     theNum = ""
18
19     # Get a number and decide which letter it represents
20     theNum = random.randint(1,3)
21     if (theNum == 1):
22         theRef = "X"      # Map 1 to X
23     elif (theNum == 2):
24         theRef = "Y"      # Map 2 to Y
25     else:
26         theRef = "Z"      # Map 3 to Z
27
28     # Get a number and convert it to a string
29     theNum = str(random.randint(1,3))
30
31     # Put the two pieces together
32     theRef = theRef + theNum # Concatenate
33
34     return (theRef)        # Send back the result
35
36 # -----
37 # Main program
38 # -----
39 cellRef = genCellRef()    # Call to function
40 print (cellRef)

```

Solution step 3:

```

1 # -----
2 # Import libraries
3 # -----
4 import random           # Needed library
5
6 # -----
7 # Global variables
8 # -----
9 cellRef = ""           # String value
10 count = 0              # Loop control
11
12 # -----
13 # Subprograms
14 # -----

```

## 1.7: Subprograms

```

15  # ==> Write the function code here
16  def genCellRef():
17      theRef = ""                # The value returned
18      theNum = ""
19
20      # Get a number and decide which Letter it represents
21      theNum = random.randint(1,3)
22      if (theNum == 1):
23          theRef = "X"           # Map 1 to X
24      elif (theNum == 2):
25          theRef = "Y"           # Map 2 to Y
26      else:
27          theRef = "Z"           # Map 3 to Z
28
29      # Get a number and convert it to a string
30      theNum = str(random.randint(1,3))
31
32      # Put the two pieces together
33      theRef = theRef + theNum    # Concatenate
34
35      return (theRef)            # Send back the result
36
37  # -----
38  # Main program
39  # -----
40
41  while (count < 10):
42      cellRef = genCellRef()      # Call to function
43      print (cellRef)
44      count = count + 1

```

Solution step 4:

```

1  # -----
2  # Import Libraries
3  # -----
4  import random                # Needed Library
5
6  # -----
7  # Global variables
8  # -----
9  cellRef = ""                 # String value
10 count = 0                     # Loop control
11 cellList = []                # Hold all cell references
12
13 # -----
14 # Subprograms

```

## 1.7: Subprograms

```

15 # -----
16 # ==> Write the function code here
17 def genCellRef():
18     theRef = ""                # The value returned
19     theNum = ""
20
21     # Get a number and decide which letter it represents
22     theNum = random.randint(1,3)
23     if (theNum == 1):
24         theRef = "X"           # Map 1 to X
25     elif (theNum == 2):
26         theRef = "Y"           # Map 2 to Y
27     else:
28         theRef = "Z"           # Map 3 to Z
29
30     # Get a number and convert it to a string
31     theNum = str(random.randint(1,3))
32
33     # Put the two pieces together
34     theRef = theRef + theNum    # Concatenate
35
36     return (theRef)            # Send back the result
37
38 # -----
39 # Main program
40 # -----
41
42 while (count < 10):
43     cellRef = genCellRef()      # Call to function
44     cellList.append (cellRef)   # Add to List
45     count = count + 1
46 print (cellList)

```

Solution step 6:

```

1 # -----
2 # Import Libraries
3 # -----
4 import random                # Needed Library
5
6 # -----
7 # Global variables
8 # -----
9 count = 0                    # Loop control
10
11
12 # -----

```

## 1.7: Subprograms

```

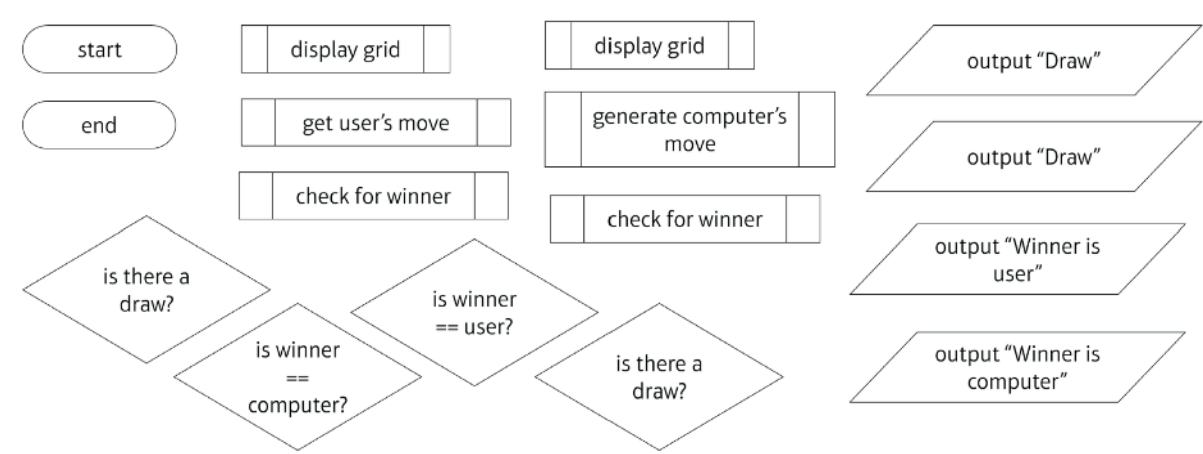
13  # Subprograms
14  # -----
15  # ==> Write the function code here
16  def genCellRef():
17      theRef = "" # The value returned
18      theNum = ""
19
20      # Get a number and decide which letter it represents
21      theNum = random.randint(1,3)
22      if (theNum == 1):
23          theRef = "X" # Map 1 to X
24      elif (theNum == 2):
25          theRef = "Y" # Map 2 to Y
26      else:
27          theRef = "Z" # Map 3 to Z
28
29      # Get a number and convert it to a string
30      theNum = str(random.randint(1,3))
31
32      # Put the two pieces together
33      theRef = theRef + theNum # Concatenate
34
35      # Print the cell reference
36      print (theRef)
37
38  # -----
39  # Main program
40  # -----
41
42  while (count < 10):
43      genCellRef() # Call to procedure
44      count = count + 1

```

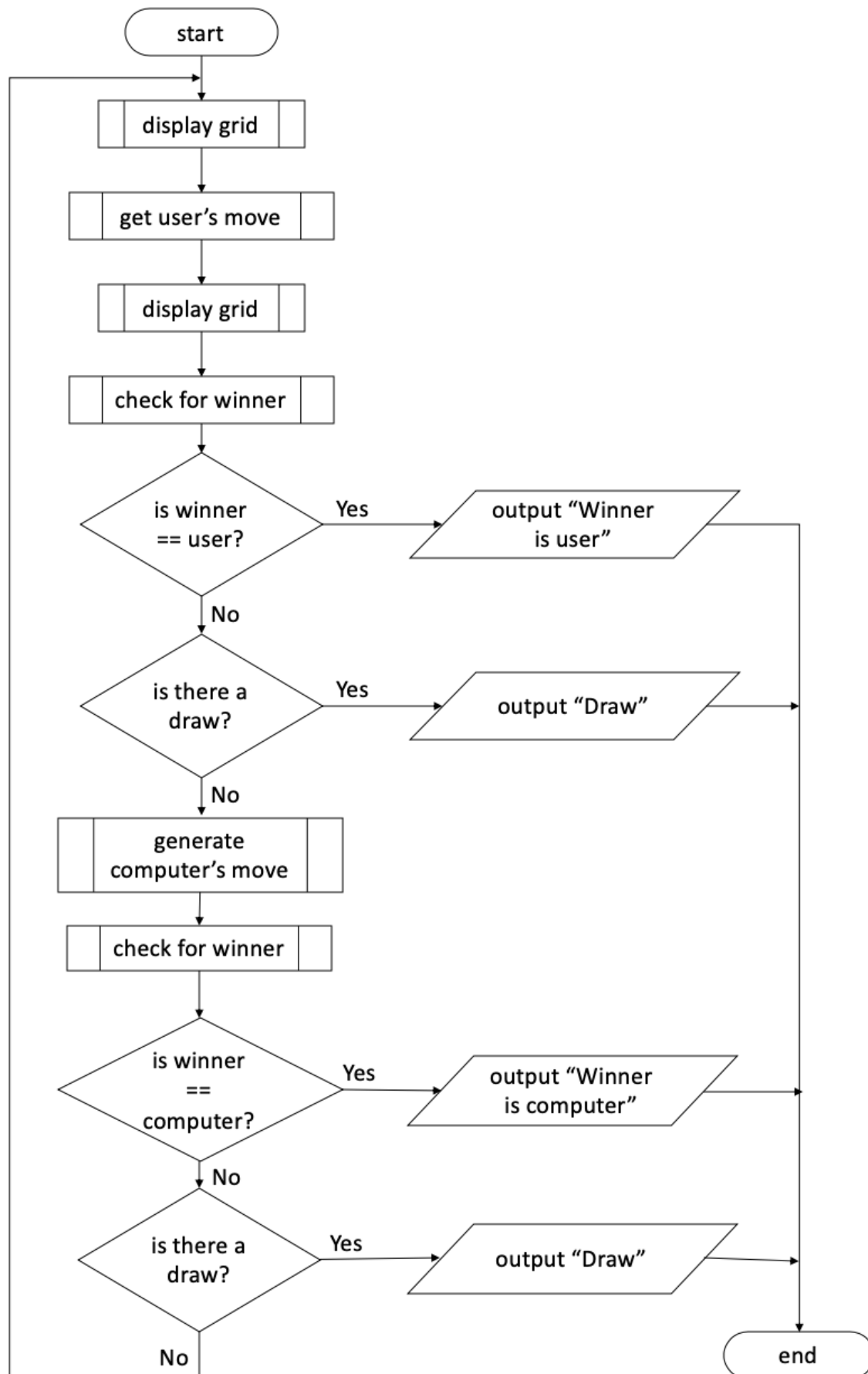
1.7: Subprograms

Activity 32 – Page 84

Another programmer has created a noughts and crosses algorithm, using a flowchart. Arrange the symbols to express an algorithm that works for noughts and crosses. Use as many arrows and Yes/No labels as required. Do not add any additional functionality or symbols.



## 1.7: Subprograms



## 1.7: Subprograms

### Activity 33 – Page 89

Practical activity using an integrated development environment. There is no solution provided.

### Activity 34 – Page 92

This program is used to calculate the sale price of a customer's shopping.

```

1  # -----
2  # Global variables
3  # -----
4  discount = 0.1          # 10% discount
5  subTotal = 0.0          # Currency is real
6  salePrice = 0.0
7  layout = ""            # For printing
8
9  # -----
10 # Subprograms
11 # -----
12 def discountPrice (total):
13     newTotal = 0.0          # Currency
14     saving = 0.0
15
16     if (total >= 50):
17         saving = total * discount
18         newTotal = total - saving
19     else:
20         newTotal = total
21
22     return (newTotal)
23
24 # -----
25 # Main program
26 # -----
27
28 subTotal = float (input ("Enter the subtotal of your shopping "))
29 salePrice = discountPrice (subTotal)
30 layout = "Sale price is £{:.2f}"
31 print (layout.format(salePrice))

```

1. Identify the local variables.  
`newTotal, saving and total`
2. Identify the global variables.  
`discount, subtotal, salePrice and layout`



## 1.7: Subprograms

- Identify a built-in subprogram.

`print()`

- Identify a user-written subprogram.

`discountPrice()`

- Identify a variable with the data type of real.

`newTotal`, `saving`, `subtotal`, `discount` and `salePrice`

- Is the subprogram `discountPrice()` a function or a procedure? How do you know?

It is a function because it uses the `return()` statement to give back a value to the calling code.

- Is the variable `total` local or global? How do you know?

It is local because all identifiers used as parameters in a subprogram definition are local to that subprogram.

### Activity 35 – Page 93

Here is the code for a lucky dip program that selects prizes for players. This program uses a built-in subprogram to convert input to uppercase.

Answer each of the questions based on the code.

```

1  # -----
2  # Import Libraries
3  # -----
4  import random
5
6  # -----
7  # Global variables
8  # -----
9  prizeList = ["Cup", "Toy", "Bracelet",
10              "T-Shirt", "Book", "Hat",
11              "Keyring", "Candle"]
12  userChoice = "Y"
13
14  # -----
15  # Subprograms
16  # -----
17  def genNumber():
18      aNumber = 0
19
20      aNumber = random.randint(0, len(prizeList) - 1)
21      return (aNumber)
22
23  def findPrize (pNum):
24      thePrize = ""
25

```

## 1.7: Subprograms

```

26     thePrize = prizeList[pNum]
27     return (thePrize)
28
29     # -----
30     # Main program
31     # -----
32     # Set up initial processing
33     print ("Welcome to the lucky dip")
34     userChoice = input ("Do you want a dip? (Y/N)")
35     userChoice = userChoice.upper()
36
37     # Loop as long as the user wants
38     while (userChoice == "Y"):
39         print ("Here we go .....")
40
41         # Figure out which prize they get
42         theNumber = genNumber()
43         thePrize = findPrize (theNumber)
44         print ("Well done you have won " + thePrize)
45
46         # Let user go again
47         userChoice = input (
48             "Do you want a dip? (Y/N)")
49         userChoice = userChoice.upper()
50
51     print ("Goodbye")

```

1. Name the local variable(s) in genNumber().

aNumber

2. Name the local variables(s) in findPrize().

pNum and thePrize

3. What do we call the role of pNum on this line (line 23)?

```
def findPrize (pNum):
```

Parameter

4. Why is the variable prizeList available when this line executes (line 26)?

```
thePrize = prizeList[pNum]
```

prizeList is a global variable, so is accessible from anywhere in the program.

5. Where is the variable aNumber created? Where is it destroyed?

Line 18 creates the local variable and line 21, the end of the function, causes it to be destroyed.

6. Where is the variable userChoice created on? Where is it destroyed?

## 1.7: Subprograms

Line 12 creates the global variable and line 51, the last line in the file, causes it to be destroyed.

7. Would this program still work if the prize list was increased to include 'Flower' and 'Candy'? Why or why not?

It would work because the length of the list is used as a parameter to the `random.randint()` function.

### Primm activity – Page 95

A program has been written to allow users to convert temperatures from Celsius to Fahrenheit.

```

1  # -----
2  # Global variables
3  # -----
4  userChoice = ""
5  userTemp = 0
6  convertedTemp = 0.0
7  layout = ""
8
9  # -----
10 # Subprograms
11 # -----
12 # Display the menu options
13 def showMenu ():
14     print ("----- Main Menu -----")
15     print ("A - Convert Celsius to Fahrenheit")
16     print ("X - Exit program")
17
18 # Get user input
19 def getUserInput ():
20     userChoice = ""
21
22     while (userChoice == ""):
23
24         # Convert whatever the user enters to upper case
25         userChoice = input ("Please make a userChoice ")
26         userChoice = userChoice.upper()
27         if (userChoice != "A") and (userChoice != "X"):
28             userChoice = ""
29             print ("Please make a valid userChoice.")
30
31     return (userChoice)
32
33 # Get user temperature
34 def getUserTemperature ():
35     return (float (input ("Please enter a temperature ")))

```

## 1.7: Subprograms

```

36
37 # Convert Celsius to Fahrenheit
38 def celsiusToFahrenheit(pTemperature):
39     # (Celsius * 9 / 5) + 32
40     return ((pTemperature * 9 / 5) + 32)
41
42 # -----
43 # Main program
44 # -----
45 print ("----- Welcome to the Temperature Conversion Program -----")
46
47 showMenu()
48 userChoice = getUserInput()
49 if (userChoice == "A"):
50     print ("Celsius to Fahrenheit")
51     userTemp = getUserTemperature()
52     convertedTemp = celsiusToFahrenheit(userTemp)
53     layout = "Original {:<7.2f} C-->F {:<7.2f}"
54     print (layout.format(userTemp, convertedTemp))
55
56 print ("Goodbye")

```

### Predict

1. What do you think the code will do?

Convert Celsius to Fahrenheit temperatures.

2. What output do you think it will produce?

Looks like there is a menu system, where the user has to type in a number and it gets converted. The format string prints the result in a decimal number.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes. It asked for a choice. I missed the X to exit the program, though.

3. Did the output match your prediction? If not, how did it differ?

I missed the X in the user menu.

### Investigate

1. How many different variables are named `userChoice`?

There are two variables named `userChoice`. One in global scope, line 4, and one in local scope, line 20.

2. Set breakpoints on every line that uses the variable `userChoice`. Execute the program using the step function of your IDE, to complete this table showing scope and content of each variable, `userChoice`.

## 1.7: Subprograms

Line number	Scope	Value
4	Main	Empty string
47	Main	Empty string
20	getUserInput	Empty string
24	getUserInput	Empty string
25	getUserInput	A
30	getUserInput	A
48	Main	A

- Consider the line: `return ((pTemperature * 9 / 5) + 32)`
- What would happen if the inside set of brackets were removed? Why?

The answers would still be calculated correctly, because all multiplication and division are done before adding.

- What would happen if adding 32 were moved to the front of the equation? Why??

Same. The order of precedence of operators is multiplication and division done first.

**Modify***Solution 1*

```

1  # -----
2  # Global variables
3  # -----
4  userChoice = ""
5  userTemp = 0
6  convertedTemp = 0.0
7  layout = ""
8
9  # -----
10 # Subprograms
11 # -----
12 # Display the menu options
13 def showMenu ():
14     print ("----- Main Menu -----")
15     print ("A - Convert Celsius to Fahrenheit")
16     print ("B - Convert Fahrenheit to Celsius")

```

## 1.7: Subprograms

```

17     print ("X - Exit program")
18
19     # Get user input
20     def getUserInput ():
21         userChoice = ""
22
23         while (userChoice == ""):
24             # Convert whatever the user enters to upper case
25             userChoice = input ("Please make a userChoice ")
26             userChoice = userChoice.upper()
27             if (userChoice != "A") and (userChoice != "B") and \
28                 (userChoice != "X"):
29                 userChoice = ""
30                 print ("Please make a valid userChoice.")
31
32         return (userChoice)
33
34     # Get user temperature
35     def getUserTemperature ():
36         return (float (input ("Please enter a temperature ")))
37
38     # Convert Celsius to Fahrenheit
39     def celsiusToFahrenheit(pTemperature):
40         # (Celsius * 9 / 5) + 32
41         return ((pTemperature * 9 / 5) + 32)
42
43     # Convert Fahrenheit to Celsius
44     def fahrenheitToCelsius(pTemperature):
45         # (Fahrenheit - 32) * 5 / 9
46         return ((pTemperature - 32) * 5 / 9)
47
48     # -----
49     # Main program
50     # -----
51     print ("----- Welcome to the Temperature Conversion Program -----")
52
53     showMenu()
54     userChoice = getUserInput()
55     if (userChoice == "A"):
56         print ("Celsius to Fahrenheit")
57         userTemp = getUserTemperature()
58         convertedTemp = celsiusToFahrenheit(userTemp)
59         layout = "Original {:<7.2f} C-->F {:<7.2f}"
60         print (layout.format(userTemp, convertedTemp))
61     elif (userChoice == "B"):
62         print("Fahrenheit to Celsius")

```

## 1.7: Subprograms

```

63     userTemp = getUserTemperature()
64     convertedTemp = fahrenheitToCelsius(userTemp)
65     layout = "Original {:<7.2f} F-->C {:<7.2f}"
66     print(layout.format(userTemp, convertedTemp))
67
68     print("Goodbye")

```

### Solution 2

```

1  # -----
2  # Global variables
3  # -----
4  userChoice = ""
5  userTemp = 0
6  convertedTemp = 0.0
7  layout = ""
8  exitProgram = False
9
10 # -----
11 # Subprograms
12 # -----
13 # Display the menu options
14 def showMenu ():
15     print ("----- Main Menu -----")
16     print ("A - Convert Celsius to Fahrenheit")
17     print ("B - Convert Fahrenheit to Celsius")
18     print ("X - Exit program")
19
20 # Get user input
21 def getUserInput ():
22     userChoice = ""
23
24     while (userChoice == ""):
25         # Convert whatever the user enters to upper case
26         userChoice = input ("Please make a userChoice ")
27         userChoice = userChoice.upper()
28         if (userChoice != "A") and (userChoice != "B") and \
29             (userChoice != "X"):
30             userChoice = ""
31             print ("Please make a valid userChoice.")
32
33     return (userChoice)
34
35 # Get user temperature
36 def getUserTemperature ():
37     return (float (input ("Please enter a temperature ")))
38

```

## 1.7: Subprograms

```

39 # Convert Celsius to Fahrenheit
40 def celsiusToFahrenheit(pTemperature):
41     # (Celsius * 9 / 5) + 32
42     return ((pTemperature * 9 / 5) + 32)
43
44 # Convert Fahrenheit to Celsius
45 def fahrenheitToCelsius(pTemperature):
46     # (Fahrenheit - 32) * 5 / 9
47     return ((pTemperature - 32) * 5 / 9)
48
49 # -----
50 # Main program
51 # -----
52 print ("----- Welcome to the Temperature Conversion Program -----")
53
54 while (not exitProgram):
55     showMenu()
56     userChoice = getUserInput()
57     if (userChoice == "A"):
58         print ("Celsius to Fahrenheit")
59         userTemp = getUserTemperature()
60         convertedTemp = celsiusToFahrenheit(userTemp)
61         layout = "Original {:<7.2f} C-->F {:<7.2f}"
62         print (layout.format(userTemp, convertedTemp))
63     elif (userChoice == "B"):
64         print("Fahrenheit to Celsius")
65         userTemp = getUserTemperature()
66         convertedTemp = fahrenheitToCelsius(userTemp)
67         layout = "Original {:<7.2f} F-->C {:<7.2f}"
68         print(layout.format(userTemp, convertedTemp))
69     elif (userChoice == "X"):
70         print("Goodbye")
71         exitProgram = True

```

### Make

A customer is visiting a restaurant. The customer wants a 3-course meal: a starter, a main and a dessert. Write a program that will suggest a dish for each course. The customer chooses 'starter', 'main' or 'dessert' from a menu of options. A random dish from that category is suggested for the customer.

In this program, use `random.randint()`, data structures for starters, mains and desserts. Use `len()` to find number of items in each data structure. Keep the number of global variables as short as possible. There should be a subprogram for each course of the meal. Keep suggesting items until the user wants to exit. Use techniques to make the program easy to understand and maintain.



## 1.7: Subprograms

*Solution*

```

1  # -----
2  # Import Libraries
3  # -----
4  import random
5
6  # -----
7  # Global variables
8  # -----
9  exitProgram = False
10 userChoice = ""
11
12 # -----
13 # Subprograms
14 # -----
15 # Display the menu options
16 def showMenu ():
17     print ("----- Meal Chooser -----")
18     print ("S - Starter course")
19     print ("M - Main course")
20     print ("D - Dessert course")
21     print ("X - Exit program")
22
23 # Get user input
24 def getUserInput ():
25     userChoice = ""
26
27     while (userChoice == ""):
28         # Convert whatever the user enters to upper case
29         userChoice = input ("Please make a userChoice ")
30         userChoice = userChoice.upper()
31         if (userChoice != "S") and (userChoice != "M") and \
32             (userChoice != "D") and (userChoice != "X"):
33             userChoice = ""
34             print ("Please make a valid userChoice.")
35
36     return (userChoice)
37
38 # Get starter course
39 def getStarter ():
40     index = 0
41     starters = ["Olives", "Garlic bread", "Soup", "Pate", "Salad"]
42     index = random.randint(0, len(starters) - 1)
43     print (index)
44
45     return (starters[index])

```

## 1.7: Subprograms

```

46
47 # Get main course
48 def getMain ():
49     index = 0
50     mains = ["Spaghetti", "Ossobuco", "Pizza", "Lasagne", "Gnocchi"]
51     index = random.randint(0, len(mains) - 1)
52     return (mains[index])
53
54 # Get dessert course
55 def getDessert ():
56     index = 0
57     desserts = ["Gelato", "Cannoli", "Panna cotta", "Tiramisu", "Pignolata"]
58     index = random.randint(0, len(desserts) - 1)
59     return (desserts[index])
60
61 # -----
62 # Main program
63 # -----
64 while (not exitProgram):
65     showMenu()
66     userChoice = getUserInput()
67     if (userChoice == "S"):
68         print ("Starter course is " + getStarter())
69     elif (userChoice == "M"):
70         print ("Main course is " + getMain())
71     elif (userChoice == "D"):
72         print ("Dessert course is " + getDessert())
73     elif (userChoice == "X"):
74         print("Goodbye")
75         exitProgram = True

```

## 1.7: Subprograms

## Exam-style questions – Page 97

1. Identify line number(s) in the code below that show one example of each of these structural components of a program. (12 marks)

```

1  # -----
2  # Import libraries
3  # -----
4  import random
5
6  # -----
7  # Global variables
8  # -----
9  carCounts = []           # Count of cars for days this week
10
11 # -----
12 # Subprograms
13 # -----
14 # Fill a data structure with a number of random numbers
15 def genNumbers (pNumberList, pCount):
16     index = 0
17
18     for index in range (0, pCount):
19         pNumberList.append (random.randint(0, 21))
20
21 # Find the maximum number in the list
22 def findMaxCount (pNumberList):
23     currentMax = -1       # Initialise to invalid number
24     index = 0
25
26     for index in range (0, len (pNumberList)):
27         if (pNumberList[index] > currentMax):
28             currentMax = pNumberList[index]
29
30     return (currentMax)
31
32 # -----
33 # Main program
34 # -----
35
36 genNumbers (carCounts, 7)
37 print (carCounts)
38 print ("The maximum count of cars is: ", findMaxCount (carCounts))

```

- a) The name of a library function

random

## 1.7: Subprograms

- b) The name of a local variable  
`index, currentMax, pNumberList, pCount`
- c) The name of a global variable  
`carCounts`
- d) The name of a data structure  
`carCounts`
- e) The name of a procedure  
`genTenNumbers`
- f) The name of a function  
`findMaxCount`
- g) The name of a parameter  
`pNumberList, pCount`
- h) The name of a built-in subprogram  
`print, len`
- i) The line number of an iteration  
`18-19, 26-28 or 18, 26`
- j) The line number of a selection  
`27-28 or 27`
- k) The line number of an initialisation  
`9, 16, 23, 24`
- l) The line number of a subprogram header  
`15, 22`

2. Referring to the code below:

```

1  # -----
2  # Global variables
3  # -----
4  area = 0.0
5  rectangleArea = 0.0
6
7  # -----
8  # Subprograms
9  # -----
10 def areaCalc(pWidth, pHeight):
11     area = 0.0
12     area = pWidth * pHeight
13     return (area)
14
15 # -----

```

## 1.7: Subprograms

```

16 # Main program
17 # -----
18
19 area = 5.0
20 rectangleArea = areaCalc (5, 10)
21 print (rectangleArea)

```

- a) State the value that will be displayed on screen after line 21 is executed. (1 mark)

50

- b) State the value of the variable area after line 12 is executed. (1 mark)

50

- c) State the value of the variable area after line 20 is executed. (1 mark)

5

- d) Explain why this is the case. (2 marks)

There are two different values because there are two different variables named area, each in different memory locations. One area (line 11) is local scope to the subprogram, while the other variable area (line 4, 19) is global to the main program.

## 1.8: Working with algorithms

### Activity 36 – Page 101

Use the following scores as test data and dry run the algorithm shown on page 99-100.

1–0 3–2 0–2 1–1

Calculate the expected end states of the variables from these results.

Track the entry of each score through the algorithm and see if they are the same as your expected results.

```

1  # -----
2  # Global variables
3  # -----
4  totalFor = 0
5  totalAgainst = 0          # Running totals
6  win = 0                  # Each game
7  loss = 0
8  draw = 0
9  goalsFor = 0             # User input
10 goalsAgainst = 0
11 anotherEntry = "Y"       # Keep looping
12 layout = \
13     "Total goals for {} and total goals against {}"
14
15 # -----
16 # Main program
17 # -----
18 while (anotherEntry == "Y"):
19     # Get the inputs
20     goalsFor = int (input ("Enter goals for: "))
21     goalsAgainst = int (input ("Enter goals against: "))
22
23     # Calculate running totals
24     totalFor = totalFor + goalsFor
25     totalAgainst = totalAgainst + goalsAgainst
26
27     # Categorise the game
28     if (goalsFor > goalsAgainst):
29         win = win + 1
30     elif (goalsFor < goalsAgainst):
31         loss = loss + 1
32     else:
33         draw = draw + 1
34
35     # Get another game
36     anotherEntry = input (
37         "Press Y to enter another result. ")

```

## 1.8: Working with algorithms

```

38     anotherEntry = anotherEntry.upper()
39
40     print (layout.format(totalFor, totalAgainst))
41     print ("Total wins: " + str(win))
42     print ("Total losses: " + str(loss))
43     print ("Total draws: " + str(draw))

```

goalsFor	goalsAgainst	totalFor	totalAgainst	win	loss	draw
1	0	1	0	1	0	0
3	2	4	2	2	0	0
0	2	4	4	2	1	0
1	1	5	5	2	1	1

Total goals for 5 and total goals against 5

Total wins: 2

Total losses: 1

Total draws: 1

### Activity 37 - Page 102

This algorithm is intended to count the number of points associated with different types of medals.

```

1  # -----
2  # Global variables
3  # -----
4  medals = ["G", "S", "B", "S", "S", "G"]
5  length = 0
6  index = 0
7  total = 0
8
9  # -----
10 # Main program
11 # -----
12 length = len(medals)
13 for index in range (0, length):
14     if (medals[index] == "G"):
15         total = total + (3 * index)
16     elif (medals[index] == "S"):
17         total = total + (2 * index)
18     else:
19         total = total + index
20 print (total)

```

## 1.8: Working with algorithms

Create and complete a trace table for this algorithm.

Length	Index	Total	Output
0	0	0	
6	0	0	
	1	2	
	2	4	
	3	10	
	4	18	
	5	33	
	6		33

### Worked example – Page 103

score	count	distinctions	merits	passes	noAward	output
0	0	0	0	0	0	
85	1	1				
75	2	2				
67	3			1		
65	4			2		
55	4		1			
0						

Test data: 85, 75, 67, 65, 55



## 1.8: Working with algorithms

Scores\_Logic\_Errors

```
Enter a score 85
Enter a score 75
Enter a score 67
Enter a score 65
Enter a score 55
Enter a score 0
Total: 5, Distinctions: 2, Merits: 1, Passes: 2, No award: 0
Goodbye
```

Scores\_No\_Errors

```
Enter a score 85
Enter a score 75
Enter a score 67
Enter a score 65
Enter a score 55
Enter a score 0
Total: 5, Distinctions: 2, Merits: 2, Passes: 1, No award: 0
Goodbye
```

### Activity 38 – Page 105

Here is an algorithm that prints out  $2^n$ . It has a logic error. Use a trace table to find and fix the logic error.

```
1  # -----
2  # Global variables
3  # -----
4  count = 0
5  stop = 2
6
7  # -----
8  # Main program
9  # -----
10
11 while (stop != 10):
12     count = count + 1
13     stop = 2 ** count
14     print ("Count is", count, "Stop is", stop)
15
16 print("Goodbye")
```

## 1.8: Working with algorithms

count	stop	output
0	2	
1	2	
2	4	
3	8	
4	16	
5	32	
6	64	

The trace table shows where the error is first spotted. The code shows the fix.

```

1  # -----
2  # Global variables
3  # -----
4  count = 0
5  stop = 2
6
7  # -----
8  # Main program
9  # -----
10
11 while (stop <= 10):
12     count = count + 1
13     stop = 2 ** count
14     print ("Count is", count, "Stop is", stop)
15
16 print("Goodbye")

```

## 1.8: Working with algorithms

## Activity 39 – Page 107

(health <= 0)	(score > 1000000)	(health <= 0) OR (score > 1000000)
0	0	0
0	1	1
1	0	1
1	1	1
0	0	0
0	1	1
1	0	1
1	1	1

(health <= 0) OR (score > 1000000)

```
if ((health <= 0) or (score > 1000000)):
    gameOver = True
```

## Activity 40 – Page 112

We could change the rules of the game in the 'game over' example to be more complex. Let's introduce a 'god' mode to change the ending of the game. The game should be over when health is 0 or lower and god mode is off. The game should also end when the score is maxed out at 1 000 000.

- Analyse the requirements.
- Construct a truth table to show all possible outcomes.

Hint: You will need 6 columns. You can calculate the number of rows based on the number of inputs. Brackets will help keep the logic organised.

godMode	(health <= 0)	(score > 1000000)	(NOT godMode)	((NOT godMode) AND (health <= 0))	((NOT godMode) AND (health <= 0)) OR (score > 1000000)
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	1	1	1

## 1.8: Working with algorithms

1	0	0	0	0	0
1	0	1	0	0	1
1	1	0	0	0	0
1	1	1	0	0	1

((NOT godMode) AND (health <= 0)) OR (score > 1000000)

```
if ((not godMode) and (health <= 0)) or (score > 1000000):
    gameOver = True
```

### Activity 41 – Page 113

A garage door opens and closes in response to a signal from a remote control in the car. The remote control has two buttons: up and down.

Write a logical expression to show when the remote will cause the door to move to the opposite state.

Translate the logical expression to program code.

door, signal

(door = open AND signal = down), (door = closed AND signal = up)

((door = open) AND (signal = down)) OR ((door = closed) AND (signal = up))

```
if (((door == OPEN) and (signal == DOWN)) or ((door == CLOSED) and
(signal == UP))):
    print ("Changing state")
```

### Activity 42 – Page 113

Lisa has built her own programmable alarm clock. She wants to program the alarm to wake her up at the correct time depending on the day.

She gets up:

- at 7.30 a.m. if it is a school day
- at 9.00 a.m. if it is the weekend or a school holiday, except on Saturdays in term time when she gets up at 8.00 a.m. to play hockey.

Write a logical expression to show the conditions to result in alarms for 7.30, 8.00, and 9.00.

Translate the logical expressions to program code.

term, day

(NOT term) OR (day = Saturday OR day = Sunday) → alarm = 0900

(term AND (day = Saturday) → alarm = 0800

otherwise → alarm = 0730

## 1.8: Working with algorithms

```
1 term = True
2 day = ""
3 alarm = ""
4
5 if (not term) or ((day == "Saturday") or (day == "Sunday")):
6     alarm = "09:00"
7 elif (term and (day == "Saturday")):
8     alarm = "08:00"
9 else:
10    alarm = "07:30"
```

## 1.8: Working with algorithms

## Exam-style questions – Page 114

1. Here is an algorithm, shown in Python, for calculating the average of two numbers. There is a logic error. Use the values of 20 and 10 for the test data.

```

1  # -----
2  # Global variables
3  # -----
4  num1 = 0          # User input
5  num2 = 0
6  average = 0.0     # Average
7  # -----
8  # Main program
9  # -----
10
11 num1 = int (input ("Enter the first number: "))
12 num2 = int (input ("Enter the second number: "))
13 average = num1 + num2 / 2
14 print ("The average of", num1, "and", num2, "is", average)

```

- a) Complete the trace table to find the logic error.

(6 marks)

num1	num2	average	output
0	0	0.0	3.
20	10	25	4.

- b) Identify the line number with the logic error.

(1 mark)

Line number 13 has the error.

- c) Write the Python code to correct the logic error.

(1 mark)

Fix for line 13. Add brackets to make sure arithmetic operations are done in the correct order. In BIDMAS, division is done before addition.

average = (num1 + num2) / 2

2. Give the output of this statement for each of the inputs:

```

if ((gender == "female") and
    ((subject == "computing") or (subject == "physics"))):
    print ("Superstar!")

```

- a) for a girl studying French

(1 mark)

No output

- b) for a girl studying computing

(1 mark)

## 1.8: Working with algorithms

Superstar!

c) for a boy studying physics.

(1 mark)

No output

3. Identify the answer that would be different if the programmer forgot the extra brackets around the OR expression. Give a reason for this difference. (2 marks)

The outputs would be nothing, Superstar!, and Supererstar!.

The reason for this is that the evaluation order without the brackets would be and followed by or. The brackets cause the or to be evaluated first, followed by the and.

## 1.9: Two-dimensional data structures

### Activity 43 – Page 119

This two-dimensional array holds the highest score achieved by each player in each of the three levels of an online game.

```
highScores = [ ["Alex", 1, 19],
                ["Seema", 1, 29],
                ["Seema", 2, 44],
                ["Lois", 1, 10],
                ["Alex", 2, 17],
                ["Alex", 3, 36],
                ["Dion", 1, 23],
                ["Emma", 1, 27],
                ["Emma", 2, 48]]
```

Construct indexing expressions to access:

- a) The three records for Alex

`highScores[0]`

`highScores[4]`

`highScores[5]`

- b) The single numbers 48 and 17

`highScores[8][2]`

`highScores[4][2]`

- c) The single number 3

`highScores[5][2]`

### Activity 44 – Page 122

1. A record data structure is to be used to store the details of music albums. Give the appropriate data type for these fields:

- a) the title of the album

`string`

- b) the name of the artist

`string`

- c) the year of release

`integer`

- d) whether the item is in stock

`Boolean`

- e) the price of the album

`real`



## 1.9: Two-dimensional data structures

f) the genre.

string

2. Create a program to work with a two-dimensional data structure for holding music album records. The program must:
  - a) declare a two-dimensional global data structure named albumTable
  - b) initialise albumTable with three records that have fields for title, artist, year of release, whether the item is in stock or not, the price and genre
  - c) create a subprogram to display the contents of albumTable in column format using string.format().

## Solution

```

1  # -----
2  # Import Libraries
3  # -----
4
5  # -----
6  # Constants
7  # -----
8
9  # -----
10 # Global variables
11 # -----
12 albumTable = [
13     ["Sunshine", "Met Office", 2018, True, 12.86, "Musicals"],
14     ["High Rise", "Builder Brigade", 2011, False, 23.76, "Classical"],
15     ["Rough Seas", "Sammy the Sailor", 1996, True, 18.34, "Folk"]]
16
17 # -----
18 # Subprograms
19 # -----
20 def displayAlbums ():
21     layout = "{:<15} {:<20} {:^4} {:^8} {:^9} {:<}"
22     print (layout.format("Title", "Artist", "Year", "In Stock", "Price", "Genre"))
23     print ("-"*75)
24
25     layout = "{:<15} {:<20} {:^4} {:^8} {:^9.2f} {:<}"
26     for album in albumTable:
27         print (layout.format(album[0], album[1], album[2],
28                               album[3], album[4], album[5]))
29
30
31 # -----
32 # Main program
33 # -----
34
35 displayAlbums()
```

## 1.9: Two-dimensional data structures

### Activity 45 – Page 125

Start this activity with the program you wrote in the activity for working with music albums (page 122). You should already have a two-dimensional data structure, with three records, and a subprogram to display the contents of the variable albumTable.

Amend the program to:

- provide a function that presents the user with a choice to (A)dd a new album, (D)isplay the whole table, or (Q)uit the program
- provide a procedure to allow the user to input the details of new albums and add that record to albumTable
- call each of the subprograms appropriately to create a functional solution, allowing the user to keep making selections until “Q” is selected.

### Solution

```

1  # -----
2  # Import Libraries
3  # -----
4
5  # -----
6  # Constants
7  # -----
8
9  # -----
10 # Global variables
11 # -----
12 albumTable = [
13     ["Sunshine", "Met Office", 2018, True, 12.86, "Musicals"],
14     ["High Rise", "Builder Brigade", 2011, False, 23.76, "Classical"],
15     ["Rough Seas", "Sammy the Sailor", 1996, True, 18.34, "Folk"]]
16
17 userChoice = ""
18
19 # -----
20 # Subprograms
21 # -----
22 def displayAlbums ():
23     layout = "{:<15} {:<20} {:^4} {:^8} {:^9} {:<}"
24     print (layout.format("Title", "Artist", "Year", "In Stock", "Price", "Genre"))
25     print ("-"*75)
26
27     layout = "{:<15} {:<20} {:^4} {:^8} {:^9.2f} {:<}"
28     for album in albumTable:
29         print (layout.format(album[0], album[1], album[2],
30                               album[3], album[4], album[5]))
31
32
33 def showMenu():

```

## 1.9: Two-dimensional data structures

```

34     key = ""
35     invalidKey = True                # Keep looping
36
37     while (invalidKey):
38
39         # Print the menu
40         print ("\n===== Menu =====")
41         print ("Please choose from the following:")
42         print ("(A)dd an album")
43         print ("(D)isplay all the albums")
44         print ("(Q)uit the program")
45
46         key = input ("What is your choice? ")
47         key = key.upper()
48
49         if (key != "A" and key != "D" and key != "Q"):
50             invalidKey = True        # Keep looping
51             print ("*** Invalid Key Try Again ***")
52         else:
53             invalidKey = False       # No Longer invalid
54
55     return (key)
56
57 def addAlbum ():
58     title = ""
59     artist = ""
60     year = 0
61     inStockString = ""
62     inStock = False
63     price = 0.0
64     genre = ""
65     record = []
66
67     # No validation is done on these input
68     title = input ("Enter the title of the album: ")
69     artist = input ("Enter the name of the artist: ")
70     year = int (input ("Enter the year of release: "))
71
72     # Convert string to corresponding Boolean
73     inStockString = (input ("If in stock, enter Y: ").upper())
74     if (inStockString == "Y"):
75         inStock = True
76     else:
77         inStock = False
78
79     price = float (input ("Enter the price: "))

```

## 1.9: Two-dimensional data structures

```

80     genre = input ("Enter the genre: ")
81
82     # Create a record
83     record.append (title)
84     record.append (artist)
85     record.append (year)
86     record.append (inStock)
87     record.append (price)
88     record.append (genre)
89
90     # Add record to table
91     albumTable.append (record)
92
93     # -----
94     # Main program
95     # -----
96
97     while (userChoice != "Q"):
98         userChoice = showMenu()
99
100        if (userChoice == "A"):
101            addAlbum()
102        elif (userChoice == "D"):
103            displayAlbums()
104
105    print ("Goodbye")

```

### PRIMM activity – Page 126

This program uses a two-dimensional data structure to hold information about mobile phones.

```

1  # -----
2  # Global variables
3  # -----
4  phoneTable = []
5
6  # -----
7  # Subprograms
8  # -----
9  def showPhones ():
10     if (len(phoneTable) != 0):
11         for phone in phoneTable:
12             print (phone)
13     else:
14         print ("Table is empty")
15

```

## 1.9: Two-dimensional data structures

```

16 def loadData (pID, pMake, pModel, pRelease, pCost):
17     aRecord = []
18     # Make a single phone record
19     aRecord.append (pID)
20     aRecord.append(pMake)
21     aRecord.append(pModel)
22     aRecord.append(pRelease)
23     aRecord.append(pCost)
24     # Append it to the phone table
25     phoneTable.append (aRecord)
26
27     # -----
28     # Main program
29     # -----
30
31     print ("\n"
32           "... Appending three mobile phone records ...")
33     loadData (18304, "SimSang", "Play", 2019, 19.00)
34     loadData (7450, "WeWaa", "Action", 2020, 36.00)
35     loadData (2421, "Peach", "Flip", 2018, 61.00)
36     showPhones()

```

### Predict

1. What do you think the code will do?

Create a data structure to hold information about mobile phones.

2. What output do you think it will produce?

The information about the phones is displayed.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes, it created a table of information.

3. Did the output match your prediction? If not, how did it differ?

Not exactly. I missed the part where the square brackets are printed as part of the table information. (This is because on line 12, the entire record is printed, not each individual field.)

### Investigate

1. What happens if the table is displayed when it is empty?

The output is only the empty square brackets : []

2. What happens if you add \n before the final quote in the information line, as shown here?

## 1.9: Two-dimensional data structures

```
print ("\n... Appending three mobile phone records ...\n")
```

There is an extra blank line before the actual data is printed.

3. Make a backup copy of the code, because the next steps will change the code.
4. What happens if the fields in one of the records to append are mixed up? You can do this by changing one of the calls to `loadData()` and mix up the inputs.

```
30 print ("\n... Appending three mobile phone records ...")
31 loadData (18304, "SimSang", "Play", 2019, 19.00)
32 loadData (7450, "WeWaa", "Action", 2020, 36.00)
33 # loadData (2421, "Peach", "Flip", 2018, 61.00)
34 # Messed up line
35 loadData ("Peach", 2018, "Flip", 2421, 61.00)
36 showPhones()
```

... Appending three mobile phone records ...

[18304, 'SimSang', 'Play', 2019, 19.0]

[7450, 'WeWaa', 'Action', 2020, 36.0]

['Peach', 2018, 'Flip', 2421, 61.0]

Is this what you want to happen?

No, this means that the fields of some records are not in the right order. It wouldn't work to process this mixed up record.

5. Restore the code from a backup copy or change the line back.
6. What happens if the variable `aRecord` is not created to hold the fields, but each field is appended directly to the `phoneTable`? You'll need to change the code in `loadData()`.

```
1 # -----
2 # Global variables
3 # -----
4 phoneTable = []
5
6 # -----
7 # Subprograms
8 # -----
9 def showPhones ():
10     if (len(phoneTable) != 0):
11         for phone in phoneTable:
12             print (phone)
13     else:
14         print ("Table is empty")
15
16 def loadData (pID, pMake, pModel, pRelease, pCost):
17     aRecord = []
```

## 1.9: Two-dimensional data structures

```

18
19     # Make a single student record
20     #     aRecord.append (pID)
21     #     aRecord.append(pMake)
22     #     aRecord.append(pModel)
23     #     aRecord.append(pRelease)
24     #     aRecord.append(pCost)
25
26     phoneTable.append (pID)
27     phoneTable.append(pMake)
28     phoneTable.append(pModel)
29     phoneTable.append(pRelease)
30     phoneTable.append(pCost)
31
32     # Append it to the phone table
33     #     phoneTable.append (aRecord)
34
35     # -----
36     # Main program
37     # -----
38
39     print ("\n... \"Appending three mobile phone records\" ...")
40     loadData (18304, "SimSang", "Play", 2019, 19.00)
41     loadData (7450, "WeWaa", "Action", 2020, 36.00)
42     loadData (2421, "Peach", "Flip", 2018, 61.00)
43     showPhones()

```

Use your IDE to set a breakpoint on the `showPhones()` call after appending the records. Run the code and use your IDE to look at the data in memory. Is this a two-dimensional data structure or a one-dimensional data structure?

```

... "Appending three mobile phone records" ...
18304
SimSang
Play
2019
19.0
7450
WeWaa
Action
2020
36.0
2421
Peach
Flip
2018

```

## 1.9: Two-dimensional data structures

61.0

This is a one-dimensional data structure. You know this because there are no square brackets used to show each individual record.

### Modify

1. Add a procedure to insert a new phone record.
2. In the main program, call the procedure to insert 5629, "Moochia", "Minus", 2019, 65.00 at index position three.
3. Add a function to find the index of the first phone that costs more than an input parameter.
4. In the main program, call the function with an input of £50.00. Print the name and model of the found phone.

### Solution (1, 2)

```

1  # -----
2  # Global variables
3  # -----
4  phoneTable = []
5
6  # -----
7  # Subprograms
8  # -----
9  def showPhones ():
10     if (len(phoneTable) != 0):
11         print ("\n")
12         for phone in phoneTable:
13             print (phone)
14     else:
15         print ("Table is empty")
16
17  def loadData (pID, pMake, pModel, pRelease, pCost):
18     aRecord = []
19
20     # Make a single student record
21     aRecord.append (pID)
22     aRecord.append(pMake)
23     aRecord.append(pModel)
24     aRecord.append(pRelease)
25     aRecord.append(pCost)
26
27     # Append it to the phone table
28     phoneTable.append (aRecord)
29
30  def insertRecord (pID, pMake, pModel, pRelease, pCost, pIndex):
31     aRecord = []
32

```



## 1.9: Two-dimensional data structures

```

33     # Make a single student record
34     aRecord.append (pID)
35     aRecord.append(pMake)
36     aRecord.append(pModel)
37     aRecord.append(pRelease)
38     aRecord.append(pCost)
39
40     # Append it to the phone table
41     phoneTable.insert (pIndex, aRecord)
42 # -----
43 # Main program
44 # -----
45
46 print ("\n... Appending three mobile phone records ...")
47 loadData (18304, "SimSang", "Play", 2019, 19.00)
48 loadData (7450, "WeWaa", "Action", 2020, 36.00)
49 loadData (2421, "Peach", "Flip", 2018, 61.00)
50 showPhones()
51 insertRecord (5629, "Moochia", "Minus", 2019, 65.00, 3)
52 showPhones()

```

*Solution (3, 4)*

```

1  # -----
2  # Global variables
3  # -----
4  phoneTable = []
5  index = -1
6
7  # -----
8  # Subprograms
9  # -----
10 def showPhones ():
11     if (len(phoneTable) != 0):
12         print ("\n")
13         for phone in phoneTable:
14             print (phone)
15     else:
16         print ("Table is empty")
17
18 def loadData (pID, pMake, pModel, pRelease, pCost):
19     aRecord = []
20
21     # Make a single student record
22     aRecord.append (pID)
23     aRecord.append(pMake)
24     aRecord.append(pModel)

```

## 1.9: Two-dimensional data structures

```

25     aRecord.append(pRelease)
26     aRecord.append(pCost)
27
28     # Append it to the phone table
29     phoneTable.append (aRecord)
30
31 def insertRecord (pID, pMake, pModel, pRelease, pCost, pIndex):
32     aRecord = []
33
34     # Make a single student record
35     aRecord.append (pID)
36     aRecord.append(pMake)
37     aRecord.append(pModel)
38     aRecord.append(pRelease)
39     aRecord.append(pCost)
40
41     # Append it to the phone table
42     phoneTable.insert (pIndex, aRecord)
43
44 def findPhoneByPrice (pPrice):
45     ndxRow = 0                # Index in table
46     found = False            # Not found
47     phoneIndex = -1          # Invalid phoneTable index
48
49     # Loop if not yet found and more records left
50     while ((not found) and (ndxRow < len(phoneTable))):
51         phone = phoneTable[ndxRow]
52
53         # Not a match, then look at next record
54         if (phone[4] <= pPrice):
55             ndxRow = ndxRow + 1
56         else:                  # Both a match
57             found = True       # Stop the loop
58             phoneIndex = ndxRow # This is the record
59
60     return (phoneIndex)
61 # -----
62 # Main program
63 # -----
64
65 print ("\n... Appending three mobile phone records ...")
66 loadData (18304, "SimSang", "Play", 2019, 19.00)
67 loadData (7450, "WeWaa", "Action", 2020, 36.00)
68 loadData (2421, "Peach", "Flip", 2018, 61.00)
69 showPhones()
70 insertRecord (5629, "Moochia", "Minus", 2019, 65.00, 3)

```

## 1.9: Two-dimensional data structures

```

71 showPhones()
72 index = findPhoneByPrice (50.00)
73 if (index != -1):
74     print ("Your phone is: ", phoneTable[index][1], phoneTable[index][2])
75 else:
76     "No phone matches criteria"

```

### Make

Create a program that keeps track of after-school clubs. Each record should hold the name of the club, the staff member in charge of the club, the day to meet and current enrolment.

A simple menu system should allow the user to display all the information held, add a new club, find names of all clubs that meet on a specific day of the week, and exit the program.

Hint: To return all the clubs, rather than just an index to a club, you need to create a new list variable in local scope and return it to the calling code.

### Solution

```

1  # -----
2  # Import Libraries
3  # -----
4
5  # -----
6  # Constants
7  # -----
8
9  # -----
10 # Global variables
11 # -----
12 clubTable = [
13     ["Drama", "Brown", "Tuesday", 24],
14     ["Martial Arts", "Green", "Thursday", 45],
15     ["Computer", "White", "Monday", 20],
16     ["Chess", "Black", "Thursday", 10]]
17 userChoice = ""                # Chosen from menu
18 club = ""
19 day = ""
20
21 # -----
22 # Subprograms
23 # -----
24 def displayClubs ():
25     layout = "{:<20} {:<15} {:<10} {:^7}"
26     print (layout.format("Name", "Staff", "Day", "Number"))
27     print ("-"*60)
28
29     layout = "{:<20} {:<15} {:<10} {:^7}"

```

## 1.9: Two-dimensional data structures

```

30     for club in clubTable:
31         print (layout.format(club[0], club[1],
32                               club[2], club[3]))
33
34
35 def showMenu():
36     key = ""
37     invalidKey = True           # Keep Looping
38
39     while (invalidKey):
40
41         # Print the menu
42         print ("\n===== Menu =====")
43         print ("Please choose from the following:")
44         print ("(A)dd a club")
45         print ("(D)isplay all the clubs")
46         print ("(F)ind a club")
47         print ("(Q)uit the program")
48
49         key = input ("What is your choice? ")
50         key = key.upper()
51
52         if (key != "A" and key != "D" and
53             key != "F" and key != "Q"):
54             invalidKey = True     # Keep Looping
55             print ("*** Invalid Key Try Again ***")
56         else:
57             invalidKey = False    # No Longer invalid
58
59     return (key)
60
61 def addClub ():
62     name = ""
63     staff = ""
64     day = ""
65     number = 0
66     record = []
67
68     # No validation is done on these input
69     name = input ("Enter the name of the club: ")
70     staff = input ("Enter the last name of the staff in charge: ")
71     day = input ("Enter the day the club meets: ")
72     number = int (input ("Enter the number of students in the club: "))
73
74     # Create a record
75     record.append (name)

```

## 1.9: Two-dimensional data structures

```

76     record.append (staff)
77     record.append (day)
78     record.append (number)
79
80     # Add record to table
81     clubTable.append (record)
82
83 def findAllByDay (pDay):
84     inday = ""
85     clubDay = ""
86     allClubs =[]
87
88     # Some validation in case one the user added does not have caps
89     inDay = pDay.upper()
90
91     for club in clubTable:
92         clubDay = club[2].upper()           # Convert to upper
93         if (clubDay == inDay):             # Found one
94             allClubs.append(club[0])
95
96     return (allClubs)
97
98
99
100
101 # -----
102 # Main program
103 # -----
104
105 while (userChoice != "Q"):
106     userChoice = showMenu()
107
108     if (userChoice == "A"):
109         addClub()
110     elif (userChoice == "D"):
111         displayClubs()
112     elif (userChoice == "F"):
113         day = input ("What day would you like to go? ")
114         theClubs = findAllByDay (day)
115         if (len(theClubs) != 0):
116             for club in theClubs:
117                 print (club)
118         else:
119             print ("No clubs meet that day. ")
120
121 print ("Goodbye")

```

## 1.9: Two-dimensional data structures

### Exam-style questions – Page 128

Private Postal People (P-Cubed) is a national delivery service specialising in small letters, large letters, small packages and medium packages.

P-Cubed use a two-dimensional data structure containing source post codes, destination post codes, and the weight of small packages in kilograms.

- a) Here is a partially complete program to find all the packages that are being sent from, or sent to, postcodes beginning with the letters “BH”.

```

1  # -----
2  # Global variables
3  # -----
4  packageTable = [["PO23", "BH17", 0.5], ["CA12", "B28", 0.23],
5                  ["HG6", "EX2", 1.2], ["HS1", "EH74", 1.15],
6                  ["PL15", "NP22", 0.75], ["SN5", "WG13", 0.62],
7                  ["PL15", "BH22", 1.33], ["BH5", "WG13", 0.75],
8                  ["EX2", "SN5", 0.8], ["NP22", "EH74", 1.15]]
9
10 foundPackages = []
11
12 # -----
13 # Subprograms
14 # -----
15
16 def findAllToAndFrom (pPlace):
17     inPlace = ""
18     allFound =[]
19
20     # =====> Write your code here
21
22     return (allFound)
23
24 # -----
25 # Main program
26 # -----
27
28 # Find and print all the packages going to and from BH
29 foundPackages = findAllToAndFrom ("BH")
30 for each in foundPackages:
31     print (each)
32
33 print ("Goodbye")

```

A console session is shown which displays the correct output.

```
['PO23', 'BH17', 0.5]
```

```
['PL15', 'BH22', 1.33]
```

## 1.9: Two-dimensional data structures

```
['BH5', 'WG13', 0.75]
```

Goodbye

Complete the code for the function `findAllToAndFrom()` to make a functional program.

(5 marks)

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  packageTable = [ ["P023", "BH17", 0.5], ["CA12", "B28", 0.23],
5                   ["HG6", "EX2", 1.2], ["HS1", "EH74", 1.15],
6                   ["PL15", "NP22", 0.75], ["SN5", "WG13", 0.62],
7                   ["PL15", "BH22", 1.33], ["BH5", "WG13", 0.75],
8                   ["EX2", "SN5", 0.8], ["NP22", "EH74", 1.15]]
9
10 foundPackages = []
11
12 # -----
13 # Subprograms
14 # -----
15
16 def findAllToAndFrom (inPlace):
17     allFound = []
18
19     # =====> Write your code here
20     for package in packageTable:
21         source = package[0]
22         destination = package[1]
23
24         if ((source[0] == inPlace[0] and source[1] == inPlace[1]) or
25             (destination[0] == inPlace[0] and destination[1] == inPlace[1])):
26             allFound.append (package)
27
28     return (allFound)
29
30 # -----
31 # Main program
32 # -----
33
34 # Find and print all the packages going to and from BH
35 foundPackages = findAllToAndFrom ("BH")
36 for each in foundPackages:
37     print (each)
38
39 print ("Goodbye")

```

## 1.9: Two-dimensional data structures

b) Here is a partially complete program to find the first overweight small package.

```

1  # -----
2  # Global variables
3  # -----
4  packageTable = [ ["P023", "BH17", 0.5], ["CA12", "B28", 0.23],
5                   ["HG6", "EX2", 1.2], ["HS1", "EH74", 1.15],
6                   ["PL15", "NP22", 0.75], ["SN5", "WG13", 0.62],
7                   ["PL15", "BH22", 1.33], ["BH5", "WG13", 0.75],
8                   ["EX2", "SN5", 0.8], ["NP22", "EH74", 1.15]]
9
10 index = 0                                # Index of found item
11
12 # -----
13 # Subprograms
14 # -----
15 def findFirstOverweight (pWeight):
16     location = -1                        # Invalid location
17
18     # =====> Write your code here
19
20     return (location)
21
22 # -----
23 # Main program
24 # -----
25
26 # Find the first overweight package
27 index = findFirstOverweight(1)
28 if (index != -1):
29     print (packageTable[index])
30 else:
31     print ("None found")
32
33 print ("Goodbye")

```

A console session is shown which displays the correct output.

```
['HG6', 'EX2', 1.2]
```

```
Goodbye
```

Complete the code for the function `findFirstOverweight()` to make a functional program. (5 marks)



## 1.9: Two-dimensional data structures

## Solution

```

1  # -----
2  # Global variables
3  # -----
4  packageTable = [
5      ["P023", "BH17", 0.5], ["CA12", "B28", 0.23],
6      ["HG6", "EX2", 1.2], ["HS1", "EH74", 1.15],
7      ["PL15", "NP22", 0.75], ["SN5", "WG13", 0.62],
8      ["PL15", "BH22", 1.33], ["BH5", "WG13", 0.75],
9      ["EX2", "SN5", 0.8], ["NP22", "EH74", 1.15]]
10
11
12 index = 0 # Index of found item
13
14 # -----
15 # Subprograms
16 # -----
17
18 def findFirstOverweight (pWeight):
19     location = -1 # Invalid location
20
21     # =====> Write your code here
22     found = False
23     index = 0 # Looping through
24
25     while (not found) and (index < len(packageTable)):
26         package = packageTable[index]
27         if (package[2] <= pWeight):
28             index = index + 1
29         else:
30             found = True
31             location = index
32
33     return (location)
34
35 # -----
36 # Main program
37 # -----
38
39 # Find the first overweight package
40 index = findFirstOverweight(1)
41 if (index != -1):
42     print (packageTable[index])
43 else:
44     print ("None found")
45
46 print ("Goodbye")

```

## 1.10: Validation and strings

## Activity 46 – Page 131

Change the code in the example on the left so that it only accepts lower- and upper-case letters, rather than numbers. Ensure you use meaningful identifiers and make your code easy to understand.

```

1  # -----
2  # Global variables
3  # -----
4  validNum = False           # Assume everything is invalid
5  userNum = 0               # Set to an invalid value
6
7  # -----
8  # Main program
9  # -----
10
11 while (not validNum):
12     userNum = int (input (
13         "Enter a number between 1 and 10 "))
14     if (userNum >= 1) and (userNum <= 10):
15         validNum = True
16     else:
17         print ("Invalid number, try again ")
18
19 print ("You entered " + str(userNum))

```

## Solution

```

1  # -----
2  # Global variables
3  # -----
4  validChar = False         # Assume everything is invalid
5  userChar = "1"           # Set to an invalid value
6
7  # -----
8  # Main program
9  # -----
10
11 while (not validChar):
12     userChar = input ("Enter a lower or upper case letter ")
13     if ((userChar >= "a") and (userChar <= "z")) or \
14         ((userChar >= "A") and (userChar <= "Z")):
15         validChar = True
16     else:
17         print ("Invalid input, try again ")
18
19 print ("You entered " + userChar)

```

## 1.10: Validation and strings

### Activity 47 – Page 132

Part of a program asks students to type in their student ID numbers. Valid ID numbers are exactly five digits long. Write a short program that demonstrates validation for ID numbers. Use a constant in your program.

Hint: Change the way you think about the problem. Just because it's an ID number does not mean you have to treat it as a number.

#### Solution

```

1  # -----
2  # Constants
3  # -----
4  LENGTHID = 5                # Constant
5
6  # -----
7  # Global variables
8  # -----
9  idNumString = ""           # Initialise to empty string
10 length = 0                 # Invalid length
11
12 # -----
13 # Main program
14 # -----
15
16 idNumString = input ("Enter your student ID: ")
17 length = len(idNumString)   # For easier reading
18
19 if (length == LENGTHID):
20     print ("Valid student ID")
21 else:
22     print ("Invalid student ID")

```

### Activity 48 – Page 133

Revisit the program you wrote that asked students to type in their student ID numbers. Recall that valid ID numbers are exactly five digits long.

- Amend the program to keep the user inside an input loop doing a presence check.
- Once the user has typed in anything, then move on to the length check.

#### Solution

```

1  # -----
2  # Constants
3  # -----
4  LENGTHID = 5                # Constant
5
6  # -----

```

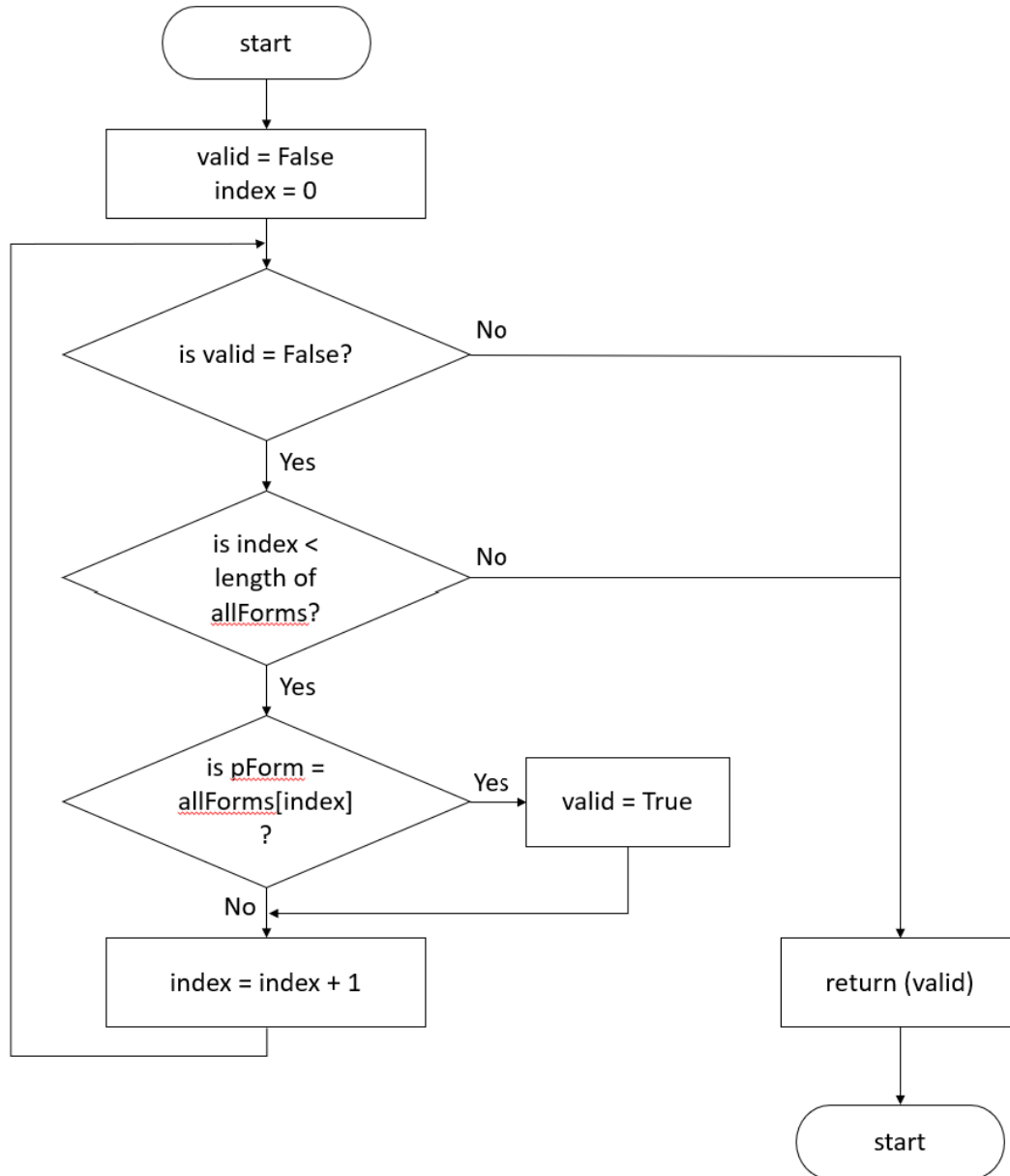
## 1.10: Validation and strings

```
7  # Global variables
8  # -----
9  idNumString = ""          # Initialise to empty string
10 length = 0                # Invalid length
11
12 # -----
13 # Main program
14 # -----
15 # While empty string, keep looping
16 while (idNumString == ""):
17     idNumString = input ("Enter your student ID: ")
18
19 length = len(idNumString)  # For easier reading
20 if (length == LENGTHID):
21     print ("Valid student ID")
22 else:
23     print ("Invalid student ID")
```

## 1.10: Validation and strings

## Activity 49 – Page 134

Draw a flowchart for the isValidForm() subprogram in the example above.



## 1.10: Validation and strings

## Activity 50 – Page 137

Change the code in the example above, to use `string.upper()`.

Test your program with these values:

6	YES	NO	Yes
No	Yes	No	£50

## Solution

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume all invalid
5  userChoice = ""              # Empty string
6
7  # -----
8  # Main program
9  # -----
10
11 while (not validChoice):
12     userChoice = input ("Enter either 'Yes' or 'No' ")
13     userChoice = userChoice.upper()
14     if ((userChoice == "YES") or (userChoice == "NO")):
15         validChoice = True
16     else:
17         print ("Invalid input, try again ")
18
19 print ("You entered " + userChoice)

```

## Activity 51 – Page 138

Amend the code in the worked example so that, regardless of whatever capitalisation the user enters, the user's first name is stored and displayed with the first letter in upper case and the remainder in lower case.

Hint: Use a combination of `string.upper()`, `string.lower()`, indexing, and a for loop with string concatenation inside it. Replace the line `newname = firstName` with your new logic. There will be more than one line.

## Solution

```

1  # -----
2  # Constants
3  # -----
4  MIN_NAME = 1                 # Constants for name length
5  MAX_NAME = 20
6
7  # -----

```

## 1.10: Validation and strings

```

8  # Global variables
9  # -----
10 firstName = ""           # User types
11 valid = False           # Assume everything is invalid
12 length = 0              # Invalid length
13 newName = ""            # Invalid capitalised name
14
15 # -----
16 # Main program
17 # -----
18 while (not valid):
19     firstName = input ("Enter your first name: ")
20     length = len(firstName)
21     if (length >= MIN_NAME) and (length <= MAX_NAME):
22         if (firstName.isalpha()):
23             firstName = firstName.lower()      # All lower
24             newName = firstName[0]             # First letter
25             newName = newName.upper()          # To cap
26
27             # Add all the remaining Letters
28             for index in range (1, len(firstName)):
29                 newName = newName + firstName[index]
30
31             print ("Your entry is valid.\nYour name is: ", newName)
32             valid = True
33         else:
34             print ("Your entry is not valid")
35     else:
36         print ("Your entry is not the right size")

```

### Activity 52 – Page 140

Line 17 is the new addition to the code in the original worked example, starting on page 139.

`formName = formName.upper()`

```

1  # -----
2  # Constants
3  # -----
4  FORM_LEN = 4             # Constants for name length
5
6  # -----
7  # Global variables
8  # -----
9  formName = ""           # User types
10 valid = False           # Assume everything is invalid
11

```

## 1.10: Validation and strings

```

12  # -----
13  # Main program
14  # -----
15  while (not valid):
16      formName = input ("Enter a form name: ")
17      formName = formName.upper()
18      if (len(formName) == FORM_LEN):          # Length is good
19          if (formName.isalnum()):             # Letters and digits only
20              # Check first character is 7, 8, or 9
21              if ((formName[0] == "7") or
22                  (formName[0] == "8") or
23                  (formName[0] == "9")):
24                  # Good first char, check remaining three
25                  if (formName[1].isalpha() and
26                      formName[2].isalpha() and
27                      formName[3].isalpha()):
28                      valid = True
29                      print ("Your entry is valid:", formName)
30              else:
31                  print ("Last three characters must be letters")
32          else:
33              print ("First character must be 7, 8, or 9")
34      else:
35          print ("Your entry is not valid")
36  else:
37      print ("Form name is not correct length")

```

### Activity 53 – Page 146

Access online documentation for Python. Find out how to change the worked example above to only handle TypeError. Change the code.

#### Solution

```

1  # -----
2  # Global variables
3  # -----
4  score = 0                                # Invalid score
5
6  # -----
7  # Main program
8  # -----
9
10 while (score == 0):                      # Keep looping
11     try:
12         # This is the normal program code. It executes.
13         score = int (input ("Enter a score: "))

```



## 1.10: Validation and strings

```

14     except TypeError:
15         # If there is any Exception this code executes
16         print ("Your entry is invalid. Try again.")
17         score = 0          # Keep Loop going
18     else:
19         # If there is no Exception, then this code executes
20         print ("You entered:", score)
21
22 print ("Goodbye")

```

Run your code and enter 'X'.

Enter a score: X

Traceback (most recent call last):

File "C:/Code/Exception\_Handling\_Specific.py", line 13, in <module>

```
score = int (input ("Enter a score: "))
```

ValueError: invalid literal for int() with base 10: 'X'

What is the effect of specifying specific exceptions to handle?

It lets all the other types of exceptions through.

### Activity 54 – Page 146

The first column in this table identifies a type of validation. For each one, provide an example of normal data, boundary data and erroneous data. Some have been done for you.

Validation	Normal	Boundary	Erroneous
Length must be at least 2 and no more than 5	"AB34"	"AB" "12345"	"A" "ABC456"
Must be a single capital letter	"R"	"A" "Z"	"a" "AB" "2"
Must be "M", "S", "X", "m", "s", "x"	"S" "s"		"R" "y"
Must be two letters followed by two digits	"AB12"	"AA00" "ZZ99"	"13GH"
Any number	123		Empty, not provided

## 1.10: Validation and strings

### PRIMM activity – Page 147

This program is part of an application that keeps track of information associated with swimming pools, such as the depth, the maximum capacity, and the colour of the tiles along the edge.

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume all input is invalid
5  userChoice = 0               # Set to an invalid value
6  exitProgram = False         # Assume no exiting
7
8  # Tile colours
9  tileColours = ["Pink", "Red", "Yellow", "Magenta", "Purple"]
10 poolDepths = [1.9, 2.4, 3.5, 4.4, 5.2]      # Water depth
11 poolCapacities = [25, 45, 20, 35, 40]      # Capacity
12
13 # -----
14 # Subprograms
15 # -----
16 def showMenu():
17     print ("Welcome to the swimming pool database.")
18     print ("-"*40)
19     print ("Option 1: Add a depth. ")
20     print ("Option 9: Exit program.")
21
22 def doAddDepth ():
23     depth = 0.0                # Invalid depth
24     depthString = ""          # Invalid string
25
26     while (depth == 0.0):
27         depthString = input ("Enter a depth, with a decimal: ")
28         try:
29             depth = float (depthString)      # Will it convert?
30         except:
31             print ("Invalid entry. Try again.") # Invalid data
32             depth = 0.0                      # Keep looping
33         else:
34             # Valid depth entered
35             poolDepths.append(depth)
36
37     print ("Depth added")
38
39 # -----
40 # Main program
41 # -----
42

```

## 1.10: Validation and strings

```

43 while (not validChoice) and (not exitProgram):
44     showMenu()
45     userChoice = int (input ("Enter an option: "))
46     if (userChoice == 1):
47         doAddDepth()           # Call depth subprogram
48     elif (userChoice == 9):
49         exitProgram = True     # Exiting
50     else:
51         print ("Invalid option, try again ")
52
53 print ("Goodbye")

```

### Predict

1. What do you think the code will do?

Shows a menu and lets the user make a choice. The user can enter a 1 or a 9.

2. What output do you think it will produce?

Looks like it only displays the menu and whether a depth is valid or not.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes, it displayed the menu and added a decimal depth.

3. Did the output match your prediction? If not, how did it differ?

The program keeps going in a loop and so does taking the depth.

### Investigate

1. What happens if a letter is entered as a menu option? Why does this happen?

ValueError: invalid literal for int() with base 10: 'k'

It happens because the int() conversion cannot work on a letter only digits.

2. What happens if an invalid option number is given, for example 5?

The 'invalid input' message is shown and the program keeps looping.

3. Change the input line to:

```
userChoice = input ("Enter an option: ")
```

What happens if you choose to exit the program? Why? Can you change other lines to make it work?

The 'invalid input' message is shown.

This is because the input string is not converted to a number.

```

if (userChoice == '1'):
elif (userChoice == '9'):

```

## 1.10: Validation and strings

## Modify

1. Add a subprogram to validate and add a tile colour; do not use try...except...else
2. Add a subprogram to validate and add capacity; do not use try...except...else

## Solution

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume everything is invalid
5  userChoice = 0               # Set to an invalid value
6  exitProgram = False         # Assume no exiting
7  poolDepths = [1.9, 2.4, 3.5, 4.4, 5.2]    # Water depth
8  # Tile colours
9  tileColours = ["Pink", "Red", "Yellow", "Magenta", "Purple"]
10 poolCapacities = [25, 45, 20, 35, 40]      # Capacity
11
12 # -----
13 # Subprograms
14 # -----
15 def showMenu():
16     print ("Welcome to the swimming pool database.")
17     print ("-"*40)
18     print ("Option 1: Add a depth. ")
19     print ("Option 2: Add a tile colour. ")
20     print ("Option 3: Add a capacity")
21     print ("Option 9: Exit program.")
22
23 def doAddCapacity ():
24     capacity = 0               # Invalid capacity
25     capString = ""            # For typing string
26
27     while (capacity == 0):
28         capString = input ("Enter an integer capacity: ")
29         if (not capString.isdigit()):
30             # Invalid value entered
31             print ("Invalid entry. Try again.")
32             capacity = 0
33         else:
34             # Valid capacity entered
35             capacity = int (capString)
36             poolCapacities.append (capacity)
37     print ("Capacity added")
38
39 def doAddDepth ():
40     depth = 0.0                # Invalid depth
41     depthString = ""           # Invalid string

```

## 1.10: Validation and strings

```

42
43     while (depth == 0.0):
44         depthString = input ("Enter a depth, with a decimal: ")
45         try:
46             depth = float (depthString)           # Will it convert?
47         except:
48             print ("Invalid entry. Try again.") # Invalid data
49             depth = 0.0                          # Keep looping
50         else:
51             # Valid depth entered
52             poolDepths.append(depth)
53
54     print ("Depth added")
55
56
57 # -----
58 # Main program
59 # -----
60
61 while (not validChoice) and (not exitProgram):
62     showMenu()
63     userChoice = int (input ("Enter an option: "))
64     if (userChoice == 1):
65         doAddDepth()           # Call depth subprogram
66     elif (userChoice == 3):
67         doAddCapacity ()       # Call capacity subprogram
68     elif (userChoice == 9):
69         print ("Goodbye")
70         exitProgram = True     # Exiting
71     else:
72         print ("Invalid option, try again ")

```

### Make

Write a program that generates and validates usernames.

Valid usernames are the first two letters of the user's last name, in upper case, followed by the number of the user's birth month (1 to 12). Examples of valid usernames are: BR10, QT12, AN5, RS9.

The program must meet these requirements:

- Have a menu system with three options, one to generate usernames, one to validate usernames, and one to exit the program
- The program should loop until the user chooses to exit the program
- There should be a subprogram to display the menu, a subprogram to generate usernames, and a subprogram to validate usernames
- Error messages should be meaningful and help the user identify why their input is invalid
- Do not use `try...except...else` for validation

## 1.10: Validation and strings

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume everything is invalid
5  userChoice = 0                # Set to an invalid value
6  exitProgram = False          # User not leaving
7
8  # -----
9  # Subprograms
10 # -----
11 def showMenu():
12     print ("Welcome to the temperature conversion program.")
13     print ("-"*50)
14     print ("Option 1: Make a user name. ")
15     print ("Option 2: Check a user name. ")
16     print ("Option 9: Exit program.")
17
18 def makeUserName ():
19     lastName = ""
20     monthString = ""
21     month = 0
22     userName = ""
23
24     while (len(lastName) == 0):
25         lastName = input("Enter the user's last name: ")
26         if (lastName.isalpha() and (len (lastName) >= 2)):
27             # Good name
28             while (len(monthString) == 0):           # Last name must be at least 2
29                 monthString = input ("Enter number of birth month: ")
30                 if (monthString.isdigit()):
31                     month = int (monthString)
32                     if ((month >= 1) and (month <= 12)):
33                         # Good month and good name
34                         lastName = lastName.upper()
35                         userName = lastName[0] + lastName[1] + str(month)
36                     else:
37                         print ("Month must be between 1 and 12.")
38                         monthString = ""
39                 else:
40                     print ("Only digits allowed in month.")
41                     monthString = ""
42             else:
43                 print ("Name must be letters only and at least 2 letters long.")
44                 lastName = ""

```

## 1.10: Validation and strings

```

45
46     return (userName)
47
48 def checkUserName ():
49     isValid = False                # Assume invalid
50     numPart = ""                  # First two characters only
51     namePart = ""                 # Chars 3 and 4
52
53     userName = input ("Enter the user name: ")
54     length = len(userName)
55     if (length >= 3) and (length <= 4):    # Possible
56         if (userName.isalnum()):          # Alphanumeric only
57             namePart = userName[0] + userName[1]
58             if (namePart.isupper()):      # Upper case 2 Letters
59                 # Calculate how many digit parts
60                 if (length == 3):        # One digit
61                     numPart = userName[2]
62                 else:
63                     numPart = userName[2] + userName[3]    # Two digits
64             # Check for all digits
65             if (numPart.isdigit()):
66                 month = int (numPart)
67                 if ((month >= 1) and (month <= 12)):
68                     isValid = True
69             else:
70                 print ("Month out of range.")
71         else:
72             print ("Must end in digits only.")
73     else:
74         print ("Characters must be upper case.")
75     else:
76         print ("Invalid character in user name.")
77 else:
78     print ("Invalid length.")
79
80     return (isValid)
81
82 # -----
83 # Main program
84 # -----
85 while (not validChoice) and (not exitProgram):
86     showMenu()
87     userChoice = int (input ("Enter an option: "))
88     if (userChoice == 1):
89         print ("User name is " + makeUserName())
90     elif (userChoice == 2):

```

**1.10: Validation and strings**

```
91         if (checkUserName()):                # All valid
92             print ("The name is valid")
93         else:
94             print ("The name is not valid")
95     elif (userChoice == 9):
96         exitProgram = True                # Exiting
97     else:
98         print ("Invalid option, try again ")
99
100 print ("Goodbye")
```



## 1.10: Validation and strings

## Exam-style questions – Page 149

1. A program has a menu with options F, L and A. A user is asked to type in an option. State the name of two different types of validation that will be needed. (2 marks)

Presence, lookup or length

2. The age of a student must be between 5 and 19, inclusive. Write the instruction to implement a range check. (3 marks)

(age >= 5) and (age <= 19)

3. Stock identifiers for widgets must follow the pattern of LLNNN. Complete the table to give one example of each type of test data. (3 marks)

Normal	AB123
Boundary	AA000, AA999, ZZ000, ZZ999
Erroneous	RST01, R, <empty>

4. Here is a program that uses the technique of keeping the user inside a loop until valid data is entered. Complete the code as indicated in the comments. (3 marks)

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume everything is invalid
5  userChoice = 0                # Set to an invalid value
6  exitProgram = False          # Assume no exiting
7
8  # -----
9  # Subprograms
10 # -----
11 def showMenu():
12     print ("Welcome to the program.")
13     print ("-"*40)
14     print ("Option 1: ")
15     print ("Option 2: ")
16     print ("Option 9: ")
17
18 # -----
19 # Main program
20 # -----
21
22 # =====> Add a WHILE loop to keep the user going around
23 #           until they make a valid choice
24
25     showMenu()
26     userChoice = int (input ("Enter an option: "))

```

## 1.10: Validation and strings

```

27     if (userChoice == 1):
28         print ("Option 1")
29     elif (userChoice == 2):
30         print ("Option 2")
31     elif (userChoice == 9):
32         exitProgram = True
33     else:
34         print ("Invalid option, try again ")
35
36 print ("Goodbye")

```

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False           # Assume everything is invalid
5  userChoice = 0               # Set to an invalid value
6  exitProgram = False          # Assume no exiting
7
8  # -----
9  # Subprograms
10 # -----
11 def showMenu():
12     print ("Welcome to the program.")
13     print ("-"*40)
14     print ("Option 1: ")
15     print ("Option 2: ")
16     print ("Option 9: ")
17
18 # -----
19 # Main program
20 # -----
21
22 # =====> Add a WHILE loop to keep the user going around
23 #         until they make a valid choice
24 while (not validChoice) and (not exitProgram):
25     showMenu()
26     userChoice = int (input ("Enter an option: "))
27     if (userChoice == 1):
28         print ("Option 1")
29     elif (userChoice == 2):
30         print ("Option 2")
31     elif (userChoice == 9):
32         exitProgram = True
33     else:

```

## 1.10: Validation and strings

```

34         print ("Invalid option, try again ")
35
36 print ("Goodbye")

```

5. An online, music-streaming site keeps track of customers and passwords. Here is the partially completed code to simulate this.

```

1  # -----
2  # Global variables
3  # -----
4  custTable = [ ["fred@somewhere.com", "poppies"],
5                ["mj28@somewhere.com", "happycat"],
6                ["redfern@somewhere.com", "brownsuit"],
7                ["patty123@somewhere.com", "bookcase"],
8                ["lifeguard@somewhere.com", "bluepool"],
9                ["wendywizard@somewhere.com", "candleflame"]]
10 customer = ""
11 password = ""
12
13 # -----
14 # Subprograms
15 # -----
16 # =====> Complete the subprogram
17 def isCustomer (          ):
18
19
20 # -----
21 # Main program
22 # -----
23 customer = input ("Enter your customer name: ")
24 password = input ("Enter your password: ")
25 # =====> Write your code here

```

Amend the code to create a functional solution that meets these requirements:

- prints a valid user message if the customer email address and password are found in the internal data structure
- prints an invalid user message if the customer email address and password are not found in the internal data structure. (15 marks)

Award marks as follows:

- 'pCust' in header (1)
- 'pPass' in header (1)
- order matches call (1)
- while loop, instead of for loop (1)
- check for found or not found (1)

## 1.10: Validation and strings

- check for length (1)
- picking up right record (1)
- comparing pCust against [0] (1)
- comparing pPass against [1] (1)
- return of Boolean value (1)
- two parameters to isCustomer in main program that match isCustomer order (1)
- check result of isCustomer with if (1)
- + LBMS Functionality

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  custTable = [ ["fred@somewhere.com", "poppies"],
5                ["mj28@somewhere.com", "happycat"],
6                ["redfern@somewhere.com", "brownsuit"],
7                ["patty123@somewhere.com", "bookcase"],
8                ["lifeguard@somewhere.com", "bluepool"],
9                ["wendywizard@somewhere.com", "candleflame"]]
10 customer = ""
11 password = ""
12
13 # -----
14 # Subprograms
15 # -----
16 # =====> Complete the subprogram
17 def isCustomer (pCust, pPass):
18     found = False
19     index = 0
20     length= 0
21
22     print ("Debugging only: ", pCust, pPass)
23     length = len (custTable)
24     while ((not found) and (index < length)):
25         if (custTable[index][0] == pCust):
26             if (custTable[index][1] == pPass):
27                 found = True
28             index = index + 1
29     return (found)
30
31 # -----
32 # Main program
33 # -----
34 customer = input ("Enter your customer name: ")
35 password = input("Enter your password: ")

```

## 1.10: Validation and strings

```
36 # =====> Write your code here
37 if (isCustomer(customer, password)):
38     print ("Valid customer")
39 else:
40     print ("Invalid customer")
```

## 1.11: Working with files

### Activity 55 – Page 154

1. Copy these lines into a file called 'TestData.txt' and save it to disk.

David Whorton,15,22.8  
 Karen White,18,25.3  
 Alex Machin,16.23.7  
 Debra Lee,17,25.2

2. In the same folder, create a program called 'TestProgram'. Copy line 1 and line 2 from the table in Figure 1.11.3 as the main program. Save the program.

```
theFile = open("ExamMarks.txt", "r")
theFile.close()
```

3. Run the program. What happens to the contents of 'TestData.txt' after running the code to open the file for reading and close the file? Does this make sense?

The contents of the file are exactly as they were before. There is no change. Yes, this makes sense because we expect the file on disk to always be there. No code was executed to overwrite or change file contents.

4. Change your program to only include line 3 and line 2 from the table in Figure 1.11.3 as the main program. Save the program.

```
theFile = open("ExamMarks.txt", "w")
theFile.close()
```

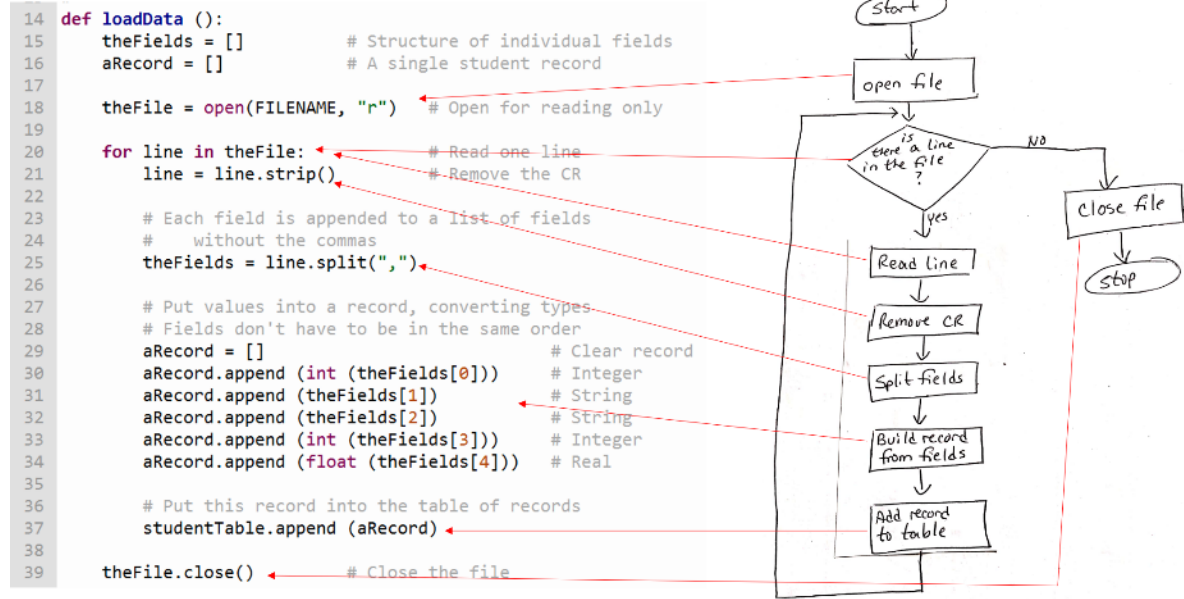
5. Run the program. What happens to the contents of 'TestData.txt' after running the code to open the file for writing and close the file? Does this make sense?

The contents of the file have disappeared. Yes, because each time a file is opened, it must be processed from the beginning.

### Activity 56 – Page 155

1. Print the flowchart (Figure 1.11.5) and the code from the loadData() subprogram in the above worked example onto a sheet of paper, side by side. Annotate the images to show how the logic of the flowchart is implemented in the code.

## 1.11: Working with files



2. Give the five lines from the worked example that will always be the same logic, regardless of how the lines are processed.

`theFile = open(FILENAME, "r")` # Open for reading only

`for line in theFile:` # Read one line  
`line = line.strip()` # Remove the CR

`theFields = line.split(",")` # Split the line into fields

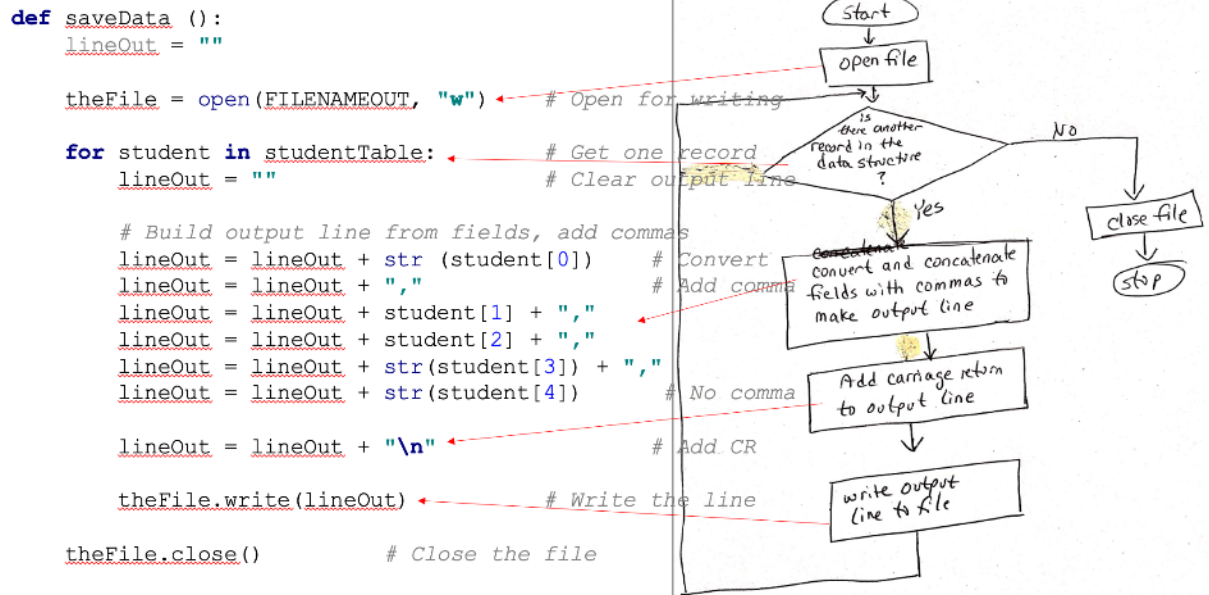
... ..

`theFile.close()` # Close the file

## 1.11: Working with files

### Activity 57 – Page 158

- Print the flowchart (Figure 1.11.6) and the code from the saveData() subprogram in the above worked example. Annotate them to show how the logic of the flowchart is implemented in the code.



- Give the five lines from the worked example that will always be the same logic, regardless of how the lines are processed.

```
theFile = open(FILENAMEOUT, "w") # Open for writing

for student in studentTable: # Get one record
    ... ..

    lineOut = lineOut + "\\n" # Add CR

    theFile.write(lineOut) # Write the line

theFile.close() # Close the file
```

### Activity 58 – Page 159

When saving records to a file, you can choose to write or append. Investigate the difference between the two. Write an appendData() subprogram and call it in the previous worked example instead of saveData().

#### Solution

```
1 # -----
2 # Constants
```



## 1.11: Working with files

```

3  # -----
4  FILENAME = "StudentRecords.txt"
5  FILENAMEOUT = "StudentRecordsOutput.txt"
6
7  # -----
8  # Global variables
9  # -----
10 studentTable = []          # Holds all the records
11
12 # -----
13 # Subprograms
14 # -----
15 def loadData ():
16     theFields = []          # Structure of individual fields
17     aRecord = []            # A single student record
18
19     theFile = open(FILENAME, "r")  # Open for reading only
20
21     for line in theFile:          # Read one line
22         line = line.strip()       # Remove the CR
23
24         # Each field is appended to a list of fields
25         # without the commas
26         theFields = line.split(",")
27
28         # Put values into a record, converting types
29         # Fields don't have to be in the same order
30         aRecord = []             # Clear record
31         aRecord.append (int (theFields[0]))  # Integer
32         aRecord.append (theFields[1])        # String
33         aRecord.append (theFields[2])        # String
34         aRecord.append (int (theFields[3]))  # Integer
35         aRecord.append (float (theFields[4])) # Real
36
37         # Put this record into the table of records
38         studentTable.append (aRecord)
39
40     theFile.close()            # Close the file
41
42 def saveData ():
43     lineOut = ""
44
45     theFile = open(FILENAMEOUT, "w")  # Open for writing
46
47     for student in studentTable:      # Get one record
48         lineOut = ""                  # Clear output line

```

## 1.11: Working with files

```

49
50     # Build output line from fields, add commas
51     lineOut = lineOut + str (student[0])    # Convert
52     lineOut = lineOut + ","                # Add comma
53     lineOut = lineOut + student[1] + ","
54     lineOut = lineOut + student[2] + ","
55     lineOut = lineOut + str(student[3]) + ","
56     lineOut = lineOut + str(student[4])    # No comma
57
58     lineOut = lineOut + "\n"                # Add CR
59
60     theFile.write(lineOut)                  # Write the line
61
62     theFile.close()                        # Close the file
63
64 def appendData ():
65     lineOut = ""
66
67     theFile = open(FILENAMEOUT, "a")       # Open for writing
68
69     for student in studentTable:           # Get one record
70         lineOut = ""                       # Clear output line
71
72         # Build output line from fields, add commas
73         lineOut = lineOut + str (student[0])    # Convert
74         lineOut = lineOut + ","                # Add comma
75         lineOut = lineOut + student[1] + ","
76         lineOut = lineOut + student[2] + ","
77         lineOut = lineOut + str(student[3]) + ","
78         lineOut = lineOut + str(student[4])    # No comma
79
80         lineOut = lineOut + "\n"                # Add CR
81
82         theFile.write(lineOut)                  # Write the line
83
84     theFile.close()                        # Close the file
85
86 # -----
87 # Main program
88 # -----
89 loadData ()                               # Load the data from file
90
91 # For testing only, change the records
92 for student in studentTable:
93     student[0] = student[0] + 100
94     student[1] = student[1].upper()

```

## 1.11: Working with files

```

95
96 saveData()           # Save the data to file
97 appendData()        # Append the changed records

```

**PRIMM activity – Page 160**

This program manipulates data about hashtags on social networking sites. The data consists of a tag name, the year it was first used, and whether it is still active.

Example lines from the data file is shown here.

```
summer,2012,T
cat,2014,T
```

The code to manipulate the data is shown here.

```

1  # -----
2  # Constants
3  # -----
4  FILENAMEIN = "HashTagsIn.txt"
5
6  # -----
7  # Global variables
8  # -----
9  tagTable = []           # Holds all the records
10
11 # -----
12 # Subprograms
13 # -----
14 def displayTags():
15     layout = "{:<20} {:<8} {:<6}"
16     tag = ""
17     active = ""
18
19     print (layout.format("Hash Tags", "Date", "Active"))
20     print ("-"*40)
21
22     for tag in tagTable:
23         if (tag[2] == True):
24             active = "Yes"
25         else:
26             active = "No"
27         print (layout.format(tag[0], tag[1], active))
28
29 def loadData ():
30     theFields = []       # Structure of individual fields

```

## 1.11: Working with files

```

31     aRecord = []                # A single record
32
33     theFile = open(FILENAMEIN, "r")    # Open for reading only
34
35     for line in theFile:            # Read one line
36         line = line.strip()        # Remove the CR
37
38         # Each field is appended to a list of fields
39         # without the commas
40         theFields = line.split(",")
41
42         # Put values into a record, converting types
43         # Fields don't have to be in the same order
44         aRecord = []                # Clear record
45         aRecord.append (theFields[0])    # String
46         aRecord.append (int (theFields[1])) # Integer
47
48         if (theFields[2] == "T"):        # Boolean
49             aRecord.append (True)
50         else:
51             aRecord.append (False)
52
53         # Put this record into the table of records
54         tagTable.append (aRecord)
55
56     theFile.close()                # Close the file
57
58 # -----
59 # Main program
60 # -----
61 loadData ()
62 displayTags ()

```

### Predict

1. What do you think the code will do?

Open a data file and read in every record to build a data table.

2. What output do you think it will produce?

The list of hashtags stored in the data structure.

### Run

1. Load the code into your coding environment and run it.
2. Did it do what you thought it would do?

Yes, it made a data table.

## 1.11: Working with files

- Did the output match your prediction? If not, how did it differ?

Yes, but the output looks like a table, and I expected it to just be the fields.

### Investigate

- In `loadData()`, the active field is converted from “T/F” to `True/False` when it is read in. Why has this been done?

As the active field is only holding two values, then it should be stored as a Boolean, to make it easy to process.

- In `displayData()`, comment out the entire IF/ELSE statement and run the program. How does the output differ from the original? Why has the IF/ELSE construct been used?

As the active field is Boolean, it prints out as 0/1, which is not very user friendly. The IF/ELSE causes that field to be converted to something more meaningful to the user.

- Use your debugger and memory inspection to answer the following questions based on an input line of “nature,2019,T”.

- What is the initial content of the variable `line`, as soon as it is read from the file?

```
'nature,2019,T\n'
```

- What is the content of the variable `line`, after `line.strip()`?

```
'nature,2019,T'
```

- What is the content of the variable `theFields`, after `line.split()`?

```
['nature','2019','T']
```

- What is the final content of the variable `aRecord`, just before it is appended to the `tagTable`?

```
['nature',2019,True]
```

### Modify

- In the main program, after loading the data, add these three records to the data structure. You can just hard code these as they're for testing only.

```
landscape,2017,T
```

```
smile,2010,F
```

```
family,2019,T
```

- Add a subprogram to create a new text file and write the contents of the data structure to it, using commas as separators.

### Solution

```
1 # -----
2 # Constants
3 # -----
4 FILENAMEIN = "HashTagsIn.txt"
```

## 1.11: Working with files

```

5  FILENAMEOUT = "HashTagsOut.txt"
6
7  # -----
8  # Global variables
9  # -----
10 tagTable = []                # Holds all the records
11
12 # -----
13 # Subprograms
14 # -----
15 def displayTags():
16     layout = "{:<20} {:<8} {:<6}"
17     tag = ""
18     active = ""
19
20     print (layout.format("Hash Tags", "Date", "Active"))
21     print ("-"*40)
22
23     for tag in tagTable:
24         if (tag[2] == True):
25             active = "Yes"
26         else:
27             active = "No"
28         print (layout.format(tag[0], tag[1], active))
29
30 def loadData ():
31     theFields = []           # Structure of individual fields
32     aRecord = []            # A single record
33
34     theFile = open(FILENAMEIN, "r")    # Open for reading only
35
36     for line in theFile:        # Read one line
37         line = line.strip()     # Remove the CR
38
39         # Each field is appended to a list of fields
40         # without the commas
41         theFields = line.split(",")
42
43         # Put values into a record, converting types
44         # Fields don't have to be in the same order
45         aRecord = []           # Clear record
46         aRecord.append (theFields[0])    # String
47         aRecord.append (int (theFields[1])) # Integer
48
49         if (theFields[2] == "T"):        # Boolean
50             aRecord.append (True)

```

## 1.11: Working with files

```

51         else:
52             aRecord.append (False)
53
54             # Put this record into the table of records
55             tagTable.append (aRecord)
56
57     theFile.close()          # Close the file
58
59 def saveData ():
60     lineOut = ""
61
62     theFile = open(FILENAMEOUT, "w")    # Open for writing
63
64     for tag in tagTable:                # Get one record
65         lineOut = ""                  # Clear output line
66
67         # Build output line from fields, add commas
68         lineOut = str (tag[0])          # Convert
69         lineOut = lineOut + ","         # Add comma
70         lineOut = lineOut + str(tag[1]) + ","
71         if (tag[2] == True):
72             lineOut = lineOut + "T"
73         else:
74             lineOut = lineOut + "F"
75
76         lineOut = lineOut + "\n"        # Add CR
77
78         theFile.write(lineOut)          # Write the line
79
80     theFile.close()          # Close the file
81
82 # -----
83 # Main program
84 # -----
85 loadData()
86 #displayTags()
87
88 # Add three records to the table (hard code)
89 tagTable.append (["smile",2010,False])
90 tagTable.append (["family",2019,True])
91 tagTable.append (["music",2015,False])
92
93 saveData()

```

## 1.11: Working with files

### Make

A text data file holds information about users of a social media platform. Each record is made up of the fields shown in this table.

Validation rules for each field are shown in the right column.

Field	Validation
Unique identifier	Between 1 and 999999, inclusive
User name	Alphabetic characters and digits only No more than 20 characters long
Number of posts	Between 0 and 999999, inclusive
Number of followers	Between 0 and 500000, inclusive

Your task is to validate the records in the file and report counts of valid records and invalid records.

Hint: Make a subprogram for each field that takes a field value as a parameter and returns true or false, depending on whether it is valid or not.

Note: in real life this file could be millions of records long. The file provided below contains only a few records that you can use for testing your program. In this data, there are five valid records and five invalid records.

0,SillySam,15231,700	4756,RainMaker,485,500001
59349,Blue=Boots,23,0	2468,FlowerPower,87,5
937,TeepeeTeepeeTeepeeTeepee,0,12	3399,FabFarmer,945,30
6462,MiniMay,2837,912	3108,PepperPot,62834,743
12,BestBaker,9582713000,489572	568,StableDoor,752,8

### Solution

```

1  # -----
2  # Constants
3  # -----
4  FILENAMEIN = "UserDataIn.txt"
5
6  # -----
7  # Global variables
8  # -----
9  countValid = 0           # Count of valid records
10 countInvalid = 0         # Count of invalid records
11 status = False          # Assume everything is invalid
12 line = ""               # Holds the line from the file

```



## 1.11: Working with files

```

13 theFields = []                # Holds the fields
14
15 # -----
16 # Subprograms
17 # -----
18 def checkUniqueID (pId):
19     idNum = 0
20     isValid = False           # Assume everything is bad
21
22     if (len(pId) != 0):       # Have something
23         if (pId.isdigit()):   # Check digits only
24             idNum = int(pId)
25                               # Range check
26             if ((idNum >= 1) and (idNum <= 999999)):
27                 isValid = True
28
29     return (isValid)
30
31
32 def checkName (pName):
33     idNum = 0
34     isValid = False           # Assume everything is bad
35
36     # Check length first
37     if ((len(pName) >= 1) and (len(pName) <= 20)):
38         if (pName.isalnum()): # Alphanumeric and digits only
39             isValid = True
40
41     return (isValid)
42
43 def checkNumPosts (pPosts):
44     numPosts = 0
45     isValid = False           # Assume everything is bad
46
47     if (len(pPosts) != 0):     # Have something
48         if (pPosts.isdigit()): # Check digits only
49             numPosts = int(pPosts)
50                               # Range check
51             if ((numPosts >= 0) and (numPosts <= 999999)):
52                 isValid = True
53
54     return (isValid)
55
56 def checkNumLikes (pLikes):
57     numLikes = 0
58     isValid = False           # Assume everything is bad

```

## 1.11: Working with files

```

59
60     if (len(pLikes) != 0):          # Have something
61         if (pLikes.isdigit()):      # Check digits only
62             numLikes = int(pLikes)
63                                     # Range check
64             if ((numLikes >= 0) and (numLikes <= 500000)):
65                 isValid = True
66
67     return (isValid)
68
69 # -----
70 # Main program
71 # -----
72 theFile = open(FILENAMEIN, "r") # Open for reading only
73
74 for line in theFile: # Read one line
75     line = line.strip() # Remove the CR
76
77     # Each field is appended to a list of fields
78     # without the commas
79     theFields = line.split(",")
80
81     # Check the unique ID field
82     status = checkUniqueID(theFields[0])
83     if (status):
84         status = checkName(theFields[1])
85         if (status):
86             status = checkNumPosts(theFields[2])
87             if (status):
88                 status = checkNumLikes(theFields[3])
89                 if (status):
90                     countValid = countValid + 1
91                 else:
92                     countInvalid = countInvalid + 1
93             else:
94                 countInvalid = countInvalid + 1
95         else:
96             countInvalid = countInvalid + 1
97     else:
98         countInvalid = countInvalid + 1
99
100 theFile.close() # Close the file
101
102 print ("Valid count =", countValid, "Invalid count =", countInvalid)

```

## 1.11: Working with files

### Exam-style questions – Page 163

Program code exists with a two-dimensional data structure already defined that holds information about different ice creams. The fields are identifier, flavour, number in stock, and cost each.

```

1  # -----
2  # Constants
3  # -----
4
5  # -----
6  # Global variables
7  # -----
8  allFood = [[10,"Vanilla",45,1.25],
9              [20,"Chocolate",52,1.32],
10             [30,"Strawberry",30,1.30],
11             [40,"Mint",15,1.45],
12             [50,"Pistachio",10,1.80],
13             [60,"Caramel",5,1.45],
14             [70,"Coconut",12,1.90],
15             [80,"Coffee",15,2.00],
16             [90,"Cherry",5,1.80],
17             [100,"Raspberry",55,1.32],
18             [110,"Mango",15,1.85]]
19
20 # -----
21 # Main program
22 # -----

```

Amend the program to meet the following requirements:

- produce one text file for all ice creams costing 1.50 or more
- produce one text file for all ice creams costing 1.49 or less
- ice creams with fewer than ten in stock should not be processed
- the program should not produce any output to the screen.

No validation of the file input is required.

(15 marks)

Open both files for writing Use for loop as processing every item Check for validity <i>before</i> doing any other work Adding commas in string Conversion of numbers to string Convert flavour to upper case	Concatenation with + Adding LF If/else selection for file >= relational operator Close both files Comments to explain logic
--	--

## 1.11: Working with files

0	1	2	3	Max.
No rewardable material	<p>Functionality (when the code is run)</p> <ul style="list-style-type: none"> <li>The component parts of the program are incorrect or incomplete, providing a program of limited functionality that meets some of the given requirements.</li> <li>Program outputs are of limited accuracy and/or provide limited information.</li> <li>Program responds predictably to some of the anticipated input.</li> <li>Solution is not robust and may crash on anticipated or provided input.</li> </ul>	<p>Functionality (when the code is run)</p> <ul style="list-style-type: none"> <li>The component parts of the program are complete, providing a functional program that meets most of the stated requirements.</li> <li>Program outputs are mostly accurate and informative.</li> <li>Program responds predictably to most of the anticipated input.</li> <li>The solution may not be robust within the constraints of the problem.</li> </ul>	<p>Functionality (when the code is run)</p> <ul style="list-style-type: none"> <li>The component parts of the program are complete, providing a functional program that fully meets the given requirements.</li> <li>Program outputs are accurate, informative, and suitable for the user.</li> <li>Program responds predictably to anticipated input.</li> <li>The solution is robust within the constraints of the problem.</li> </ul>	3

### Solution

```

1  # -----
2  # Constants
3  # -----
4  HIGHFILE = "IceCreamExpensive.txt"
5  LOWFILE = "IceCreamCheap.txt"
6
7  # -----
8  # Global variables
9  # -----
10 allFood = [[10,"Vanilla",45,1.25],
11             [20,"Chocolate",52,1.32],
12             [30,"Strawberry",30,1.30],
13             [40,"Mint",15,1.45],
14             [50,"Pistachio",10,1.80],
15             [60,"Caramel",5,1.45],
16             [70,"Coconut",12,1.90],
17             [80,"Coffee",15,2.00],
18             [90,"Cherry",5,1.80],
19             [100,"Raspberry",55,1.32],

```

## 1.11: Working with files

```

20         [110,"Mango",15,1.85]]
21
22     # -----
23     # Main program
24     # -----
25
26     highFile = open (HIGHFILE, "w")
27     lowFile = open (LOWFILE, "w")
28
29     for iceCream in allFood:          # Process them all
30         if (iceCream[2] > 5):        # Only if more than 5 in stock
31             # Build output string with commas
32             outline = ""
33             outline = outline + str(iceCream[0]) + ","
34             outline = outline + iceCream[1] + ","
35             outline = outline + str(iceCream[2]) + ","
36             outline = outline + str(iceCream[3])
37             outline = outline + "\n"          # Line feed
38
39             # Decide which file to write it to
40             if (iceCream[3] >= 1.50):
41                 highFile.write(outline)
42             else:
43                 lowFile.write(outline)
44
45     highFile.close()
46     lowFile.close()

```

## 1.12: Sorting and searching

### Activity 59 – Page 167

An array of items found in nature is shown in the table, along with index positions. Apply the bubble sort algorithm to sort the array into alphabetical order. Count the number of comparisons and swaps on each pass. Complete the table to show what happens to the array on each pass. How many passes were needed to sort the entire array?

0	1	2	3	4	Compares	Swaps	Pass
Lake	Grass	Tree	Rock	Flower	–	–	–
Grass	Lake	Rock	Flower	Tree	4	3	1
Grass	Lake	Flower	Rock	Tree	4	1	2
Grass	Flower	Lake	Rock	Tree	4	1	3
Flower	Grass	Lake	Rock	Tree	4	1	4
Flower	Grass	Lake	Rock	Tree	4	0	5

### Activity 60 – Page 167

- Here is a list of numbers [1, 2, 3, 4, 5]. Work through the bubble sort algorithm to see how many passes are needed to sort the list, in descending order. You may find it useful to draw a table, as in the previous activity.

Descending order = Five passes are needed

Ascending order = Only one pass is needed.

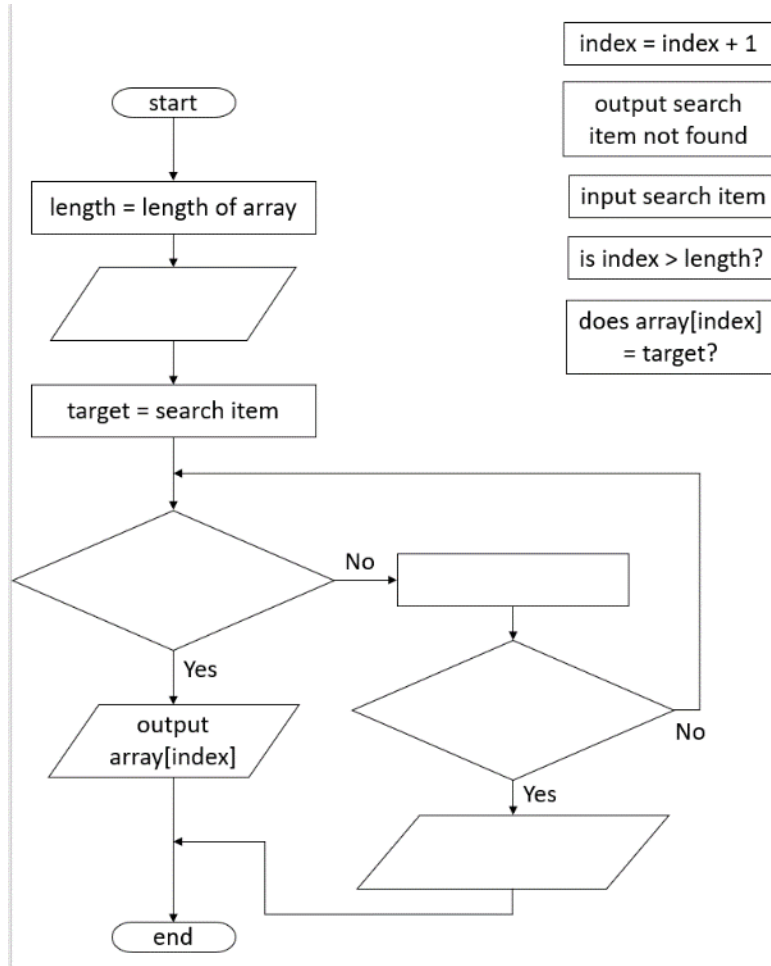
- Here is a list of numbers [5, 4, 3, 2, 1]. Work through the bubble sort algorithm to see how many passes are needed to sort the list, in ascending order. You may find it useful to draw a table, as in the previous activity.

Five passes are needed.

## 1.12: Sorting and searching

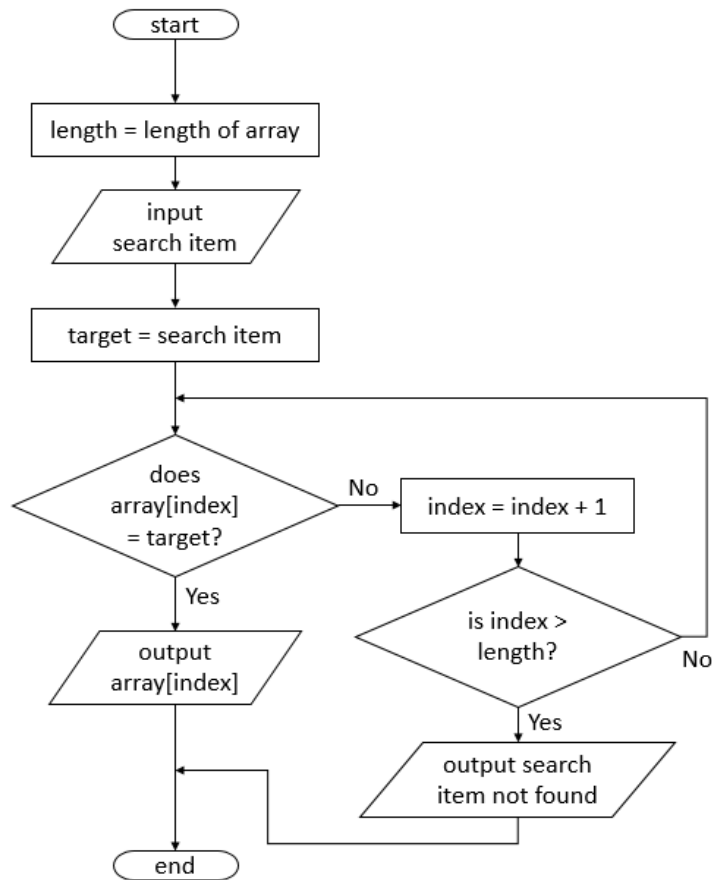
### Activity 61 – Page 171

This flowchart displays a linear search algorithm, but some of the symbols have not been labelled. Draw and complete the flowchart using the labels provided.



## 1.12: Sorting and searching

*Solution*



### Activity 62 – Page 173

Display the stages of a binary search, as in the worked example above, to find the number 13 in this list.

3 9 13 15 21 24 27 30 36 39 42 54 69

27

15

9

13



1.12: Sorting and searching

Exam-style questions – Page 175

- Here is an array of colours. A linear search is to be used to locate a target colour, but the code has not been written yet.

Explain one process that could be performed on the data that could impact the efficiency of the linear search algorithm. (2 marks)

The data should be sorted because this would allow the linear search algorithm to identify when it goes past where the target should be located.

- Here is an array of numbers. A binary search is to be used to locate the target 2048. Complete the table to show the number visited and the calculation of the median. The first has been done for you. (5 marks)

1	2	4	8	16	32	64	128	256	512	1024	2048	4096
---	---	---	---	----	----	----	-----	-----	-----	------	------	------

Visited	Median
64	$(0+12)//2=6$
512	$(7+12)//2 = 9$
2048	$(10+12)//2 = 11$

- Here is code for a bubble sort algorithm.

Amend the code to put the lines in the correct order to produce a functioning program. (10 marks)

```

1 # -----
2 # Global variables
3 # -----
4 # -----
5 # Subprograms
6 # -----
7 # -----
8 # Main program
9 # -----
10 i = 0
11 if pArray[j] > pArray[j + 1]:
12 j = 0
13 print (office)
```

## 1.12: Sorting and searching

```

14 pArray[j] = pArray[j + 1]
15 bubbleSort(office)
16 print (office)
17 temp = ""
18 for i in range(length):
19     length = len(pArray)
20     length = 0
21     temp = pArray[j]
22     def bubbleSort(pArray):
23         pArray[j + 1] = temp
24         for j in range(0, length - i - 1):
25             office = ["stapler", "ruler", "pen", "marker", "rubber", "pencil"]

```

A console session for the correct program is shown here.

```
['stapler', 'ruler', 'pen', 'marker', 'rubber', 'pencil']
```

```
['marker', 'pen', 'pencil', 'rubber', 'ruler', 'stapler']
```

### Solution

```

1  # -----
2  # Global variables
3  # -----
4  office = ["stapler", "ruler", "pen",
5            "marker", "rubber", "pencil"]
6
7  # -----
8  # Subprograms
9  # -----
10 def bubbleSort(pArray):
11     i = 0
12     j = 0
13     temp = ""
14     length = 0
15
16     length = len(pArray)
17
18     for i in range(length):
19         for j in range(0, length - i - 1):
20
21             if pArray[j] > pArray[j + 1]:
22                 temp = pArray[j]
23                 pArray[j] = pArray[j + 1]
24                 pArray[j + 1] = temp
25
26  # -----

```

## 1.12: Sorting and searching

```
27 # Main program
28 # -----
29 print (office)
30 bubbleSort(office)
31 print (office)
```

## Topic 2: Data

### Activity 1 – Page 179

1. What comes after 0000 1111?  
0001 0000 (16)
2. What comes after 1111 0111?  
1111 1000 (248)
3. What comes before 1111 0000?  
1110 1111 (239)

### Activity 2 – Page 180

1. Calculate:
  - a) how many distinct binary patterns can be produced with 8 bits  
256
  - b) how many bits are needed to represent each pupil in a class of 34 with a unique binary pattern  
6 (100010)
  - c) how many bits are needed to represent the 20 teams in a premier league.  
5 (10100)
2. Write an arithmetic expression to show that 4096 unique product codes can be represented in 12 bits.  
 $4096 = 2^{12}$

### Activity 3 – Page 181

Convert these positive binary numbers to denary.

0111 1001

121

0000 0110

6

1010 1010

170

### Activity 4 – Page 182

Convert these denary numbers to 8-bit binary.

69

0100 0101

193

## 2.1: Binary

1100 0001

12

0000 1100

### Activity 5 – Page 183

Add together these pairs of 8-bit unsigned binary numbers.

0001 1010 + 1101 0111

1111 0001

0000 1101 + 1010 1010

1011 0111

1101 0111 + 0000 1010

1110 0001

### Activity 6 – Page 184

Show the result of adding these two 8-bit binary numbers.

0110 1101 + 1100 0000

1 0010 1101. This causes an overflow error.

What could the consequences be and what can a programmer do to avoid this happening?

This could cause the program to crash or cause errors in future calculations. A programmer can avoid this happening by increasing the storage of the program to cope with additional bits. (This could be done by changing the data type from one that uses eight bits to one that uses 16 bits to store data.)

### Activity 7 – Page 186

1. What is the place value of the MSBs of these two's complement binary numbers?

a) 1011

–8

b) 1000 1100

–128

c) 1000 1000 1110

–2048

2. Convert these two's complement binary numbers to denary.

a) 1100 1100

–52

b) 0110 1101

109

c) 1000 1001

## 2.1: Binary

–119

d) 1100

–4

e) 0111

7

3. Add together these two two's complement numbers: 0010 1010 and 1101 0110.

Ignoring the overflow, what result do you get?

The result is 1 0000 0000, which is –256. Ignoring the overflow gives 0000 0000.

Why do you think this is the case?

These numbers represent +42 and –42. They add to zero. It does this with an overflow bit, which is not stored in the result.

### Activity 8 – Page 186

1. Convert these unsigned binary numbers into their two's complement negative equivalent.

a) 0001 1010

–26 1110 0110

b) 0101 0111

–87 1010 1001

c) 0010 0100

–36 1101 1100

d) 0000 1000

–8 1111 1000

2. Convert these negative denary numbers into two's complement.

a) –113

1000 1111

b) –14

1111 0010

c) –90

1011 0110

d) –42

1101 0110

### Activity 9 – Page 187

Carry out these logical binary shifts.

- a) Shift 0000 0111 two positions to the left.

## 2.1: Binary

0001 1100

- b) Shift 0011 1010 three positions to the left.

1101 0000

- c) Shift 1001 1101 one position to the right.

0100 1110

- d) Shift 1000 1110 three positions to the right.

0001 0001

### Activity 10 – Page 187

Use logical shifts to:

- a) Multiply 0001 0100 by 0010.

Equivalent to shift one position left

0010 1000 (40)

- b) Divide 0100 0110 by 0100.

Equivalent to shift two positions right

0001 0001 (17)

- c) Multiply 0000 1010 by  $2^3$ .

Equivalent to shift three positions left

0101 0000 (80)

- d) Divide 0001 0000 by  $2^2$ .

Equivalent to shift two positions right

0000 0100 (4)

### Activity 11 – Page 189

Perform the indicated arithmetic shifts on these binary patterns:

- a) Shift 1001 0000 right by 0100.

1111 1001 (shift right four times, preserve the MSB)

- b) Shift 1111 0110 left by 0001.

1110 1100 (shift left one time, fill with zero)

- c) Shift 1100 0000 right by 1000.

1111 1111 (shift right eight times, preserve the MSB)

- d) Shift 0000 1111 left by 0010.

0011 1100 (shift left by two, fill with zero)

### Activity 12 – Page 191

- Convert these binary numbers to hexadecimal.

## 2.1: Binary

a) 0100 1111

4F

b) 1001 1011

9B

c) 0010 1101

2D

2. Convert these hexadecimal numbers to binary.

a) B6

1011 0110

b) 9E

1001 1110

c) 2D

0010 1101



## 2.1: Binary

## Exam-style questions – Page 191

1. Complete the table to show how +11 is represented in binary and how -11 is represented in two's complement. (2 marks)

+11	0	0	0	0	1	0	1	1
-11	1	1	1	1	0	1	0	1

2. The bit pattern 1101 0001 uses two's complement representation. Convert this bit pattern to a denary number. (1 mark)

-47

3. Complete the table by adding these two binary numbers. (1 mark)

0	1	0	1	0	0	1	1
0	1	0	0	1	0	1	0
1	0	0	1	1	1	0	1

Alternative solution:

0	1	0	1	0	0	1	1
0	1	0	0	1	0	1	0
1	0	0	1	1	1	0	1
Carry 1					Carry 1		

4. Describe why right shifts can produce imprecise results. (2 marks)

Sometimes a binary shift produces an imprecise result. This can happen when the pattern is interpreted as numbers. For example, if we shift 0010 0001 (+33) by 1 position right (a division by 2) the rightmost bit is lost, producing a result of +16, which is clearly imprecise.

5. Explain the effect of left shifting a binary pattern that represents a positive integer. (2 marks)

Shifting a binary pattern left will multiply the number by 2 on each shift.

6. Give the result of a right logical shift by two positions on the pattern 0101 1100.

(1 mark)

0001 0111

## 2.2: Data representation

### Activity 13 – Page 193

Write down the ASCII binary codes for the characters in 'ASCII code'.

Symbol	Code
A	01000001
S	01010011
C	01000011
I	01001001
I	01001001
<space>	00100000
c	01100011
o	01101111
d	01100100
e	01100101

Alternative:

01000001 01010011 01000011 01001001 01001001 00100000 01100011 01101111  
01100100 01100101

### Activity 14 – Page 194

1. The German language includes the characters ä, ö and ü. Explain why these cannot be represented in ASCII.

With just 7 bits, ASCII cannot generate enough binary patterns to represent all the letters and symbols in common use across the world. An 8-bit version known as 'extended ASCII' attempted to overcome this problem, but even using 256 binary patterns does not enable it to support a large enough character set. (You might have heard the term Unicode. This method of representing data allow many more characters to be used.)

2. The letter 'M' has the ASCII binary code 0100 1101. Derive the code for the letters 'Q' and 'L'. (Do not look them up in the table.)

Q is four further down the alphabet than M, so add four to binary pattern.

L is one position prior to M, so subtract one from the binary pattern

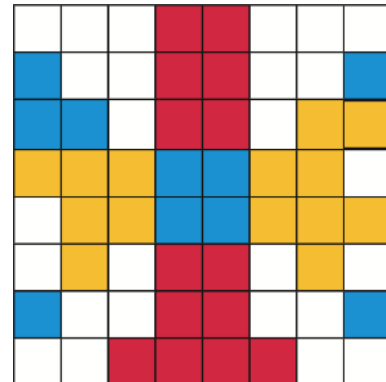
Q: 01010001 and L: 01001100

## 2.2: Data representation

## Activity 15 – Page 195

Generate the binary code for this image which has a colour depth of two.

- 00 = white
- 01 = blue
- 10 = red
- 11 = yellow



Start at the top left-hand corner, move left to right along each row, ending at the bottom right-hand corner. Ignore row ends. Where the pixel at the end of one row is the same as the pixel at the start of the row below, code them as one block.

00	00	00	10	10	00	00	00
01	00	00	10	10	00	00	01
01	01	00	10	10	00	11	11
11	11	11	01	01	11	11	00
00	11	11	01	01	11	11	11
00	11	00	10	10	00	11	00
01	00	00	10	10	00	00	01
00	00	10	10	10	10	00	00

## Activity 16 – Page 196

1. An image is 300 pixels high and 200 pixels wide.

Construct an expression to calculate how many pixels are used to represent the image.

$$\text{Total pixels} = 200 \times 300$$

2. Construct an expression to calculate the file size in kibibytes of:

- a) an 8-bit colour image with a size of  $640 \times 480$  pixels

$$\text{File size} = (640 \times 480 \times 8) / (8 \times 1024)$$

- b) a 24-bit true colour image with a size of  $640 \times 480$  pixels.

$$\text{File size} = (640 \times 480 \times 24) / (8 \times 1024)$$

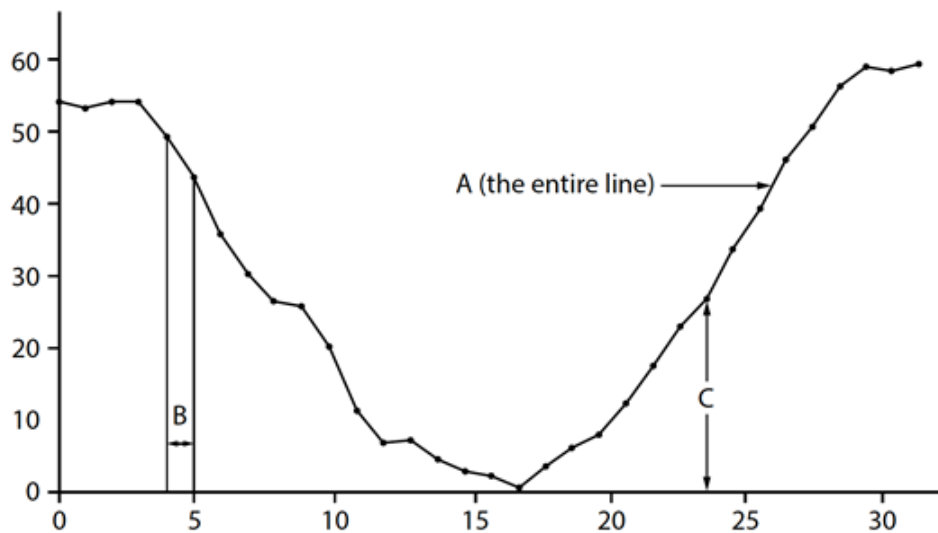
## 2.2: Data representation

- An image has a file size of 460 800 bytes, its height is 640 pixels and its width is 480 pixels. Construct an expression to calculate the colour depth of the image.

$$\text{Colour depth} = (460\,800 \times 8) / (640 \times 480)$$

### Activity 17 – Page 200

- This diagram shows the process of converting sound from analogue to digital.



- Label the x and y axes

X: time

Y: amplitude

- Give the names of the items labelled A, B and C.

A: waveform or analogue signal

B: sample interval

C: sample

### Activity 18 – Page 200

- Construct an expression to calculate the file size of a recording of three minutes duration with a sample rate of 44 100 and a bit depth of 24 bits.

$$\text{File size} = \text{sample rate} \times \text{bit depth} \times \text{time (seconds)}$$

$$\text{File size} = 44\,100 \times 24 \times 180$$

- The sample rate of a file is 30 kHz. Its file size is 5 MiB and its bit depth is 16 bits.

Construct an expression to calculate how long the sound will play.

$$\text{Duration in seconds} = \text{File size in bits} / \text{sample rate in bits per second} / \text{bit depth}$$

$$5 \text{ MiB in bits} = (5 \times 1024 \times 1024) \times 8$$

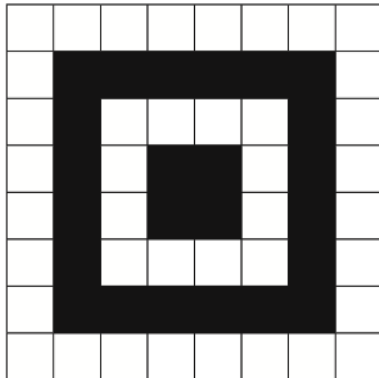
$$\text{Duration in seconds} = (5 \times 1024 \times 1024 \times 8) / (30 \times 1000 \times 16)$$

$$\text{Alternative: } (5 \times 1024 \times 1024 \times 8) / (30 \times 1000) / 16$$

## 2.2: Data representation

## Exam-style questions – Page 201

1. This diagram shows a black and white image consisting of 64 pixels.



- a) Define the term 'pixel'. (1 mark)

Pixel is short for 'picture element', the smallest element of a bitmap image.

- b) State how many bits per pixel would be needed if the image was in 16 colours rather than 2. (1 mark)

4 bits per pixel would represent 16 colours ( $2^4 = 16$ ).

- c) Explain how colour depth affects the appearance and file size of an image. (3 marks)

As the number of pixels and the colour depth increase, so too does the amount of data that has to be stored and manipulated, increasing the file size. The appearance of the image will be better as the colour depth increases, although increasing colour depth beyond 24 bits for a displayed image may not be necessary as the difference may not be seen by a human.

2. An analogue sound signal needs to be digitised.

- a) State the aspect of the analogue signal that will be sampled and stored as binary data during the analogue to digital conversion. (1 mark)

Amplitude

- b) State how reducing the bit depth affects the digital representation of the original audio. (1 mark)

If the bit depth is too low, the recording is not accurate and a lot of quiet sounds are lost.

- c) Explain how *increasing the sampling interval* affects the digital representation of the original audio. (2 marks)

The higher the *sampling interval* (time between samples), the less accurate the digital representation will be, because movements in the signal will be missed.

On the other hand, the higher the *sampling rate* (more samples per second), the more accurate the digital representation will be (although a digital representation can never completely match the original analogue sound).

## 2.2: Data representation

3. An analogue sound is never fully reproducible in digital format. Explain why this is the case. (2 marks)

Analogue sound is natural and is made of waveforms. Capturing these waveforms digitally is achieved using sampling. Because samples are taken at intervals, some of the original analogue sound will be lost between sample intervals. Analogue signals are continuous, which means they can have an infinite number of values. Computers cannot represent infinite values, as data storage and processing capacity is finite.

4. An audio sound is sampled at 24 kHz. 8 bits are used to store each sample. It is 3 minutes in duration.

Construct an arithmetic expression to determine the file size for this audio in GiBs.

(2 marks)

Do not carry out the calculation.

File size =  $(24 \times 1000 \times 8 \times 180) / (8 \times 1024 \times 1024 \times 1024)$

## 2.3: Data storage and compression

### Activity 19 – Page 202

1. A hard disk has a storage capacity of 1.5 TiB.

Express this in:

- a) mebibytes

$$1.5 \times 1024 \times 1024$$

- b) kibibytes.

$$1.5 \times 1024 \times 1024 \times 1024$$

2. An image file has a size of 363 143 213 bits. Construct an expression to give its size in mebibytes.

$$\text{Size in mebibytes} = 363\,143\,213 / (8 \times 1024 \times 1024)$$

### Activity 20 – Page 202

1. An SD card has a capacity of 32 GiB.

- a) How many 6 MiB photos can be stored on it?

$$(32 \times 1024) / 6 = 5461$$

- b) What is its capacity in bytes?

$$32 \times 1024 \times 1024 \times 1024 = 34\,359\,738\,368 \text{ bytes}$$

2. A USB memory stick has a capacity of 2 TiB. It is used to store voice recordings. If the average size of a recording is 13 MiB, how many recordings will fit on the memory stick?

$$2 \times 1024 \times 1024 = 2\,097\,152 \text{ MiB}$$

$$2\,097\,152 / 13 = 161\,319 \text{ recordings}$$

$$\text{Alternate: } (2 \times 1024 \times 1024) / 13$$

## 2.3: Data storage and compression

## Exam-style questions – Page 204

1. Amend this list by putting the units in ascending order from smallest to largest.

(1 mark)

bit	tebibyte	byte	nibble	mebibyte	kibibyte	gibibyte
-----	----------	------	--------	----------	----------	----------

bit → nibble → byte → kibibyte → mebibyte → gibibyte → tebibyte

2. A sound file has a sample rate of 96 kHz and a bit depth of 24 bits. It is 3 minutes long.

Construct an expression to determine the amount of storage needed for this file in mebibytes.

Do not carry out the calculation.

(2 marks)

File size =  $(96000 \times 24 \times 180) / (8 \times 1024 \times 1024)$

3. File size is an important consideration.

- a) Give **two** reasons for reducing file sizes.

(2 marks)

Smaller files take up less space on storage devices.

Smaller files are quicker to load because they require fewer read operations.

Smaller files are quicker to transfer over a network because fewer bytes are transmitted.

- b) One characteristic of lossy compression is that it makes files smaller.

Give **one** other characteristic of lossy compression.

(1 mark)

It removes superfluous data that is usually not detectable by humans, either sound or image.



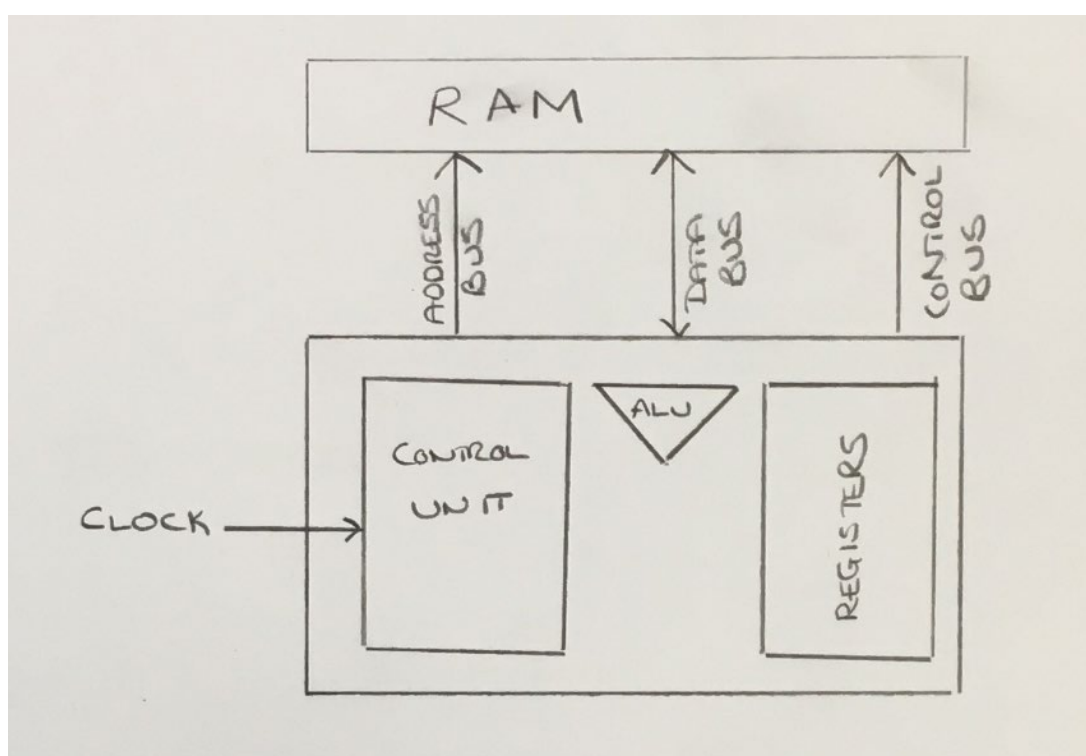
## Topic 3: Computers

### Activity 1 – Page 209

1. Explain in your own words why the stored program concept was such an important breakthrough.

The stored program concept was an important breakthrough because it allowed computers to be designed and created in such a way that program instructions and data were stored in main memory together. This meant that computers could become multipurpose, rather than a single-use device that would need to be redesigned each time.

2. Draw a diagram of the inside of a computer. Label the components and briefly describe what each of them does



- RAM (random access memory): a collection of storage locations
- Buses:
  - Address: holds the address of the memory location involved in the instruction
  - Data: holds the value to be read from or written to memory
  - Control: carries signals from the control unit to other components to coordinate operations
- Clock: sends electrical signals at regular intervals so that operations can be synchronised
- Control unit: decodes instructions received from main memory and provides a signal to coordinate operations

## **3.1: Hardware**

- Arithmetic logic unit: performs arithmetic and logical operations
- Registers: provide fast, temporary storage, e.g. registers for designated functions, such as holding the results of operations, holding instructions, and holding other data

### **Activity 2 – Page 210**

1. Make a cartoon strip or storyboard of the fetch-decode-execute cycle.

Students own cartoon strip or storyboard

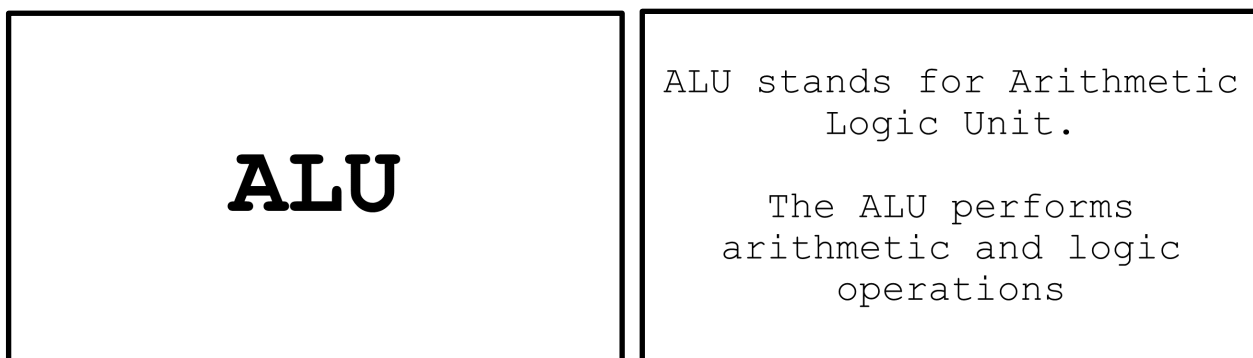
### 3.1: Hardware

2. Create a set of flashcards to help learn the terminology from this section.

The front of the card should contain the key term to define. You can find these words in bold and coloured in this section.

The back of the card should contain the definition. You can find the definitions in the Glossary and they are also explained in this text. You should write the definition in your own words. If there are words in the definition given in the Glossary that you do not understand, then look them up in a dictionary or on a computing-related online dictionary. Make a definition that is really clear to you and that you can fully understand.

Example of a flashcard for ALU:



**3.1: Hardware****Activity 3 – Page 213**

1. Main memory is sometimes referred to as primary storage.

How does primary storage differ from secondary storage?

Main memory (or primary storage) is volatile and does not retain data when the computer is powered down. Secondary storage is non-volatile, meaning data and other programs are retained when the computer is powered down.

2. Magnetic, optical and solid state are all technologies that provide permanent storage for programs and data.
  - a) Draw up a table to summarise how data is stored on magnetic, optical and solid-state devices.

Storage type	How data is stored
Magnetic	Data is stored on platters. These platters have tiny areas that can be magnetised. A magnet changes the polarity of these areas. Magnetised areas are 1; demagnetised areas are 0.
Optical	Data is stored in a series of pits and lands on the surface of a disk. A laser is used to burn the surface. Land is 1; pit is 0.
Solid-state	Data is stored on NAND flash and is made up of transistors. Transistors trap electrons in pools. Full pool is 0; empty pool is 1.

**3.1: Hardware**

- b) Add information about the capacity of, and typical uses for, each type of storage to the table.

Storage type	Capacity	Typical use
Magnetic	Gibibytes, Tebibytes (often called Gigabytes and terabytes)	It is the main storage device in most computer systems. Examples include hard disks and tapes.
Optical	80 MB to 50GB	It is used to store HD movies and other HD recordings. Examples include CD, DVD, and Blu-ray disks.
Solid-state	MB to TB	Memory cards are used as a convenient and portable removable storage medium. They are also used as the main storage device in some devices, including mobile phones, and laptop computers.

### 3.1: Hardware

#### Activity 4 – Page 214

1. Make a list of the functions carried out by a microcontroller embedded within a washing machine. What are its inputs and outputs?

##### Functions of a microcontroller in a washing machine

- Temperature control
- Time control
- Spin speed
- Drainage control
- Additional settings for garment type (sportswear, baby clothes, woollen, etc.)
- Door lock
- Child lock settings

##### Inputs

- Heat setting
- Time setting
- Spin setting
- Wash-type selection

##### Outputs

- Display settings
- Lights for selected options
- Warning and completion beeps

2. One drawback of embedded systems is that they are difficult to repair or upgrade. Why do you think this is the case?

Embedded systems are difficult to repair or upgrade because they may be fitted deep within a piece of hardware, potentially causing more damage when attempting to access them. Even if an embedded system is easy to access, the embedded system itself is small and inexpensive, so it may be more cost-effective to replace it.

### 3.1: Hardware

#### Activity 5 – Page 214

1. Printers and microwaves can be embedded systems. What input and output devices do they use?

	Input device	Output device
<b>Printer</b>	Menu button On/off button	Low-ink warning light Power control mechanism
<b>Microwave</b>	Time control dial Power level control dial Door sensor	Time counter display Power level display Light Beeper

2. A contactless smart card is an embedded system. Answer these questions about a contactless smart card.

- What is a smart card?

A smart card is a card that contains an embedded microprocessor and memory, for example, a contactless bank card.

- What additional components does a smart card need to enable contactless communication?

It needs a way to carry out wireless connectivity so it can communicate with a card reader either by physical coming together (insert the card into the reader) or by just getting near enough to the reader (near field communications).

- Where does a smart card get its power from?

Power for the smart card is received from the device it is interacting with, for example, a card reader or cash machine.

- How much data can be stored on a smart card?

A small amount: up to 8 KB of RAM and 300 KB of ROM

- How is data on a smart card secured?

Data cannot be accessed from the card's microprocessor until the card and the external device communicate with each other.

#### Exam-style questions – Page 217

1. Describe what is meant by the 'stored program concept'. (2 marks)

The stored program concept refers to a design of a computer architecture by John von Neumann. His design allowed for programming instructions and data to be stored together in main memory, thus making a computer multipurpose.

2. The central processing unit (CPU) has an important role in the von Neumann architecture.

- a) State the purpose of the CPU. (1 mark)

The purpose of the CPU is to carry out any processes that need to be carried out in a computer.

- b) State the name of the component of the CPU that decodes instructions fetched from memory. (1 mark)

Control unit (CU)

3. The hardware components of a computer system work together to carry out the fetch-decode-execute cycle.

- a) State the name of the component that synchronises their activities. (1 mark)

Control unit

Note: the clock signal feeds into the CU, which in turn coordinates the other components.

- b) Describe the role of the control unit (CU), arithmetic logic unit (ALU), the registers, the address bus and the data bus in the fetch-decode-execute cycle. (5 marks)

Fetch: The address bus holds the address in RAM of the next instruction. The control unit sends a signal, which causes the contents of the address (on the address bus) to be loaded (read) from RAM. The content (instruction) is placed on the data bus during the read.

Decode: Back in the CPU, the instruction is loaded into a register and decoded by the control unit. Once decoded, the control unit determines if the instruction needs the ALU. If so, then the required data is put into registers.

Execute: The control unit signals the ALU or other components to act and the results are put into registers.

4. Von Neumann recognised that computers need some form of secondary storage

- a) Explain why secondary storage is needed. (2 marks)

Secondary storage is required to permanently store data and programs that are required for use on a computer. This data and the programs are not lost when the power is switched off.

- b) Describe how data is physically stored on an optical disk. (2 marks)



### 3.1: Hardware

An optical disk stores data as a series of pits and lands on the surface of the disk. A land is equal to 1, and the pit is 0. A laser is used to burn the pits and lands.

- c) Activity trackers and other portable devices use solid-state storage.

Give **one** reason why solid-state storage is suitable for this purpose. (1 mark)

Solid-state storage is suitable for portable devices because it contains no moving parts and can store many gibibytes of data on a small scale.

5. A train company uses ticket vending machines at each station. The machines use embedded systems.

- a) Explain one benefit of using an embedded system in these machines. (2 marks)

Embedded systems are small, cheap to produce and replace when necessary, and are programmed for a specific purpose, for example, printing train tickets when a payment has been authorised.

- b) Customers use a touch screen to select their destination. They can pay by cash or bank card. Their tickets and a receipt are printed. The touch screen is controlled by an embedded system.

Give two other hardware components in the ticket machine that are controlled by embedded systems. (2 marks)

The help button on a ticket machine could be controlled by an embedded system, alerting the help centre employee that someone needs assistance. In addition, a speaker could be controlled by an embedded system, providing support for customers with visual impairments. The payment transaction would be completed by an embedded system that could communicate with a bank. The printer would be an embedded system that could function, once the ticket is downloaded, without interacting with the bigger system.

6. One example of an embedded system in a greenhouse is sprinklers that switch on automatically when the moisture level of the soil is too dry.

Explain how an embedded system could control the temperature of a greenhouse. (2 marks)

Temperature sensors in the greenhouse could control motors to open the vents in the roof. When the temperature is too high, the sensors would activate the motors to open the vents in the roof. When the temperature drops to an acceptable level, the motors would activate to close the vents in the roof.

7. An IoT light bulb can be switched on using an app on a mobile phone.

Explain how this is achieved. (3 marks)

The IoT light bulb is programmed to upload data to a remote server, which is then accessed by the user via a smartphone app. This means it can be remotely controlled, for example, switching it on and off, or controlling the dimmer setting.

## 3.2: Software

### Activity 6 – Page 219

Sally and Ahmed work for a company that makes printers. Ahmed writes the user manuals for the printers. Sally programs the robots that assemble the printers.

Decide what levels of access they should be given to files containing:

- designs for new printers
- user manuals for printers already on the market
- programs to control the robots
- first drafts of a user manual for a new printer to be launched later in the year.

#### Sally (Programmer): levels of access table

	Read	Write	Execute	Delete
Designs for new printers	✓	Sally is not a designer of the printer hardware		
User manuals for printers already on the market	Sally would only need this if she were using one of the printers herself			
Programs to control the robots	✓	✓	✓	✓
First drafts of a user manual for a new printer to be launched later in the year	✓ To be able to review the contents	✓ To correct errors and add comments as a reviewer		

Note: Some of these are attributed to interpreting the scenario.

### 3.2: Software

Ahmed (user manual author): levels of access table

	Read	Write	Execute	Delete
Designs for new printers	✓So that he knows how the parts work together			
User manuals for printers already on the market	✓To see the layouts and convention so any new manual matches	✓If he was updating the manuals to a new version		
Programs to control the robots				
First drafts of a user manual for a new printer to be launched later in the year	✓To be able to review the contents	✓ To correct errors and add comments as a reviewer		✓

Note: Some of these are attributed to interpreting the scenario.

## 3.2: Software

## Activity 7 – Page 219

A text file is stored on a hard drive. The file holds information from one side of a sheet of paper.

The sheet of paper is represented as a grid, 80 columns wide and 66 rows long.

Each cell in the grid contains a single 7-bit ASCII character.

The hard drive allocates space in blocks of 1024 bytes.

Construct an expression to show the number of blocks required to store the file. You do not need to do the calculation.

$\lceil (80 \times 66 \times 7) \div (8 \times 1024) \rceil$

Note: Students are not expected to use the ceiling ( $\lceil \rceil$ ) symbols, but should realise that blocks cannot be subdivided. Therefore, the output of the expression will need rounding up to the nearest whole number. The `math.ceil()` function is included in the Programming Language Subset. Integer division would not be accurate as it would effectively round down.

Expression	Result	Division type
$(80 \times 66 \times 7) \div (8 \times 1024)$	4.51171875	Regular division
$\lceil (80 \times 66 \times 7) \div (8 \times 1024) \rceil$	5	Ceiling applied to result of regular division
$(80 \times 66 \times 7) // (8 \times 1024)$	4	Integer division

## 3.2: Software

### Activity 8 – Page 220

Draw up a table to compare the benefits and drawbacks of the three scheduling algorithms mentioned above.

Scheduling algorithm	Benefits	Drawbacks
First in, first out	Completes tasks in the order in which they were requested. Every task will eventually be run.	May not complete more important processes when needed. Some tasks may have to wait a long time, if behind others that take a long time.
Shortest job first	Completes smaller jobs quickly, clearing them from CPU time allocation.	May not complete more important processes when needed. Some jobs may have to wait a long time, if there are lots of smaller ones that jump ahead.
Round robin	Each task given a slice of time depending on its priority	May not complete more important processes when needed.

### Activity 9 – Page 223

Here is a diagram of a hard disk on which five files (A, B, C, D and E) are stored. Each file is split up into several blocks. Redraw the diagram to show the arrangement of the blocks after a disk defragmenter has been run.

B3	D3	D1	C1	A4	E3	B4	D2	A2	E2	B2	C2	A1	D4	C3	E1	A3	B1
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

After disk defragmentation:

A1	A2	A3	A4	B1	B2	B3	B4	C1	C2	C3	D1	D2	D3	D4	E1	E2	E3
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Note: The order of the files may be different, i.e. C may be before A, but the order of the blocks within each file should be numerical.

### Activity 10 – Page 223

Create more flashcards to add to your collection for this topic.

See example for Activity 2, section 3.1 Hardware.

#### Activity 11 – Page 225

1. How does modular testing help keep code secure?

Modular testing improves code security because each part, e.g. module, library, file, is extensively tested in isolation before the final product is produced. This means that any inconsistencies or vulnerabilities are identified early rather than when a coded solution is complete, which would make identification of errors more difficult and time consuming.

2. 'Cyclomatic complexity' is a software metric used to measure the complexity of code. Find out how software metrics help improve the quality of code.

Software metrics improve the quality of code because they are designed to provide quality assurance, code management, debugging and performance information. Software metrics help software developers reach the more important goals of the project, such as planning, productivity, and scalability.

#### Exam-style questions – Page 226

1. The operating system (OS) manages the computer resources.
  - a) Explain how the OS uses scheduling to share use of the CPU between processes. (2 marks)

The OS uses scheduling to allocate processor time by multitasking. This involves allocating a share of the CPU, main memory and peripherals, and keeping track of their progress. Some processes may be paused to give others processor time, but will be reactivated when it is their turn again.

- b) Describe how the operating system organises files on a hard drive. (3 marks)

The operating system of a computer organises and monitors file location using a hierarchical tree structure. This means there is a root directory that contains users, directories, subdirectories and files. When a new file is stored, the operating system divides it into blocks and stores them in an empty sector on the hard drive, which can then be retrieved later. It is the OS that provides the functionality to save, open, copy, duplicate, rename and delete files because it keeps track of where all of an individual file's block are located on the disk.

2. A company stores sensitive statistics about its performance on a server. Users do not all have the same access to files stored on the server. Explain the type of access a student on work experience at the company should be given to the statistics file. (2 marks)

A student on work experience with the company would likely be inexperienced in that area of work and would also most likely be working there for only a short time. Therefore, it would be appropriate for the student to have only **read access** to the statistics file, meaning he or she could view the file, but not be able to make any amendments or delete the file.

3. Utility tools help to maintain and optimise computer performance.
  - a) Explain how a defragmenter tool can help improve the performance of a computer. (2 marks)

Performance on a computer can be improved by completing a disk defragmentation process. The computer achieves this by rearranging the scattered blocks of each into sequential order. File blocks that are scattered across the hard disk surface slows down the computer's performance.

- b) Give **two** reasons for using a data compression tool. (2 marks)
    - Compressing a data file, for example audio or video, frees up storage space on a device.
    - Compressing a data file allows faster transfer speeds across a network.
  - c) Explain why it is important to keep anti-malware up to date. (2 marks)

It is important to keep anti-malware software up to date on a device because new malware causes a regular threat. The anti-malware updates contain the newest

### 3.2: Software

files needed to combat new malware and protect the device. New malware is released every day, so the signature file needs to be regularly updated.

4. Explain the purpose of a code review. (2 marks)

The purpose of code review is to identify instances of poor programming practice, find vulnerabilities in the code and check efficiency during the development stage.

5. Making do with a temporary fix rather than taking time to find a robust solution is bad programming practice. Explain why this is the case. (2 marks)

A temporary fix to a problem may have a long-term negative impact on the solution. This kind of programming practice is likely to result in a program having insecurities, or failing as a solution all together.



## 3.3: Programming languages

## Activity 12 – Page 228

Complete this table to compare features of high-level and low-level languages.

	High-level	Low-level
<b>Readability</b>	Easily readable by a human because they use English words, but have to be translated into machine code for computers to execute.	More difficult for humans to read and write because binary patterns look similar. Assembly language is a bit easier because it uses mnemonics, a shorthand for instructions.
<b>Portability</b>	Solutions programmed in a high-level language are written independently of a particular computer architecture, meaning it is likely to be portable to different system architectures.  Note: providing that there is a language translator for both architectures.	Because low-level languages are written at a microprocessor level, they are not portable to different system architectures.
<b>Tools</b>	They are supported by environments and tools to help code solutions, including editors, syntax analysers, and debuggers.	Assembly languages may be supported by some tools, but they may be very basic and lack advanced features.
<b>Uses</b>	They support development of a wide range of solutions from device drivers to interface to hardware to high-end graphics packages. They are general purpose.	They are microprocessor specific, so are generally more suited to low-level applications, such as device drivers, embedded systems, and operating systems. When a solution needs to execute quickly, low-level is a good choice.
<b>Speed</b>	Most programmers will write code quicker in a high-level language because it is easier to read, understand, and there is tool support.	It is usually slower to code in a low-level language because it is written in binary, meaning writing and error-checking in 1s and 0s is incredibly difficult and time consuming. Assembly language makes this easier as it uses mnemonics.

### 3.3: Programming languages

<b>Ease of use</b>	A high-level language is easier to learn, write and debug. Environments in which coding takes place will include tools that highlight errors and debugging solutions.	A low-level language is harder to learn, write and debug. Working at the lowest level of detail (1s and 0s) is incredibly challenging to use.
--------------------	---	---

### 3.3: Programming languages

#### Exam-style questions – Page 230

1. Explain the role of a translator. (2 marks)

A translator takes the code written in a human-readable language, either high-level or assembly, and converts it into machine code, binary.

2. Give two reasons why high-level programming languages are preferred for most programming tasks. (2 marks)

- High-level languages are usually quicker to code in as they are written in English-like words rather than in mnemonics or binary patterns.
  - High-level languages are not dependent on any one computer architecture, so they are transferable between systems, as long as there is a language translator for the target system.
3. Manjit is writing software for a new set-top receiver for satellite TV. Explain why she should use a compiled language rather than an interpreted language for this purpose. (2 marks)

Code running on a set-top receiver must run as quickly as possible to give the customer a good viewing experience.

Manjit should use a compiled language instead of an interpreted language because the source code of the software would be translated all at once to produce an executable that can be sent to the customer. The customer does not see the source code and execution can take place without translation.

An interpreted language, which translates and executes one line at a time, would require the source code and a translator to be installed on the set-top receiver. The source code may be viewed by the customer. Interpretation each time execution is required is a slow process.

## **Topic 4: Networks**

### **Activity 1 – Page 232**

Create a leaflet explaining:

- what a network is
- the services available on your school network
- the top five reasons for connecting devices on a network

[Students own leaflet](#)

### 4.1: Networks

#### Activity 2 – Page 233

A hotel chain with its headquarters in London has 25 hotels located in cities throughout Europe. Each hotel has its own LAN and all the hotels' LANs are connected to form a WAN. Frequent guests to any of the hotels in the chain are offered a loyalty card enabling them to earn points and cash them in for additional overnight stays.

- Explain how the WAN enables the hotel chain to operate its loyalty card scheme.

A WAN (wide area network) connects the hotel chain's computers and other devices in multiple sites across Europe. This WAN means that all hotel sites are connected, meaning a customer's details of their stay can be stored centrally, and loyalty card information can be accessed at any one site.

- How does the hotel chain benefit from having a WAN?

The hotel chain benefits in having all of the sites connected, so that all data can be stored centrally but also accessed for data analysis purposes and to track trends in guest behaviour. It might also allow the chain to have a website where guests can book rooms, check bookings, and administer their own loyalty card accounts.

- How do guests benefit?

The guests of the hotel chain will benefit because they will have loyalty points automatically added to their accounts without the need to re-enter their personal information each time they visit a hotel. As all data is stored centrally, they would also be able to access their account information to check their point balance and cash in their rewards at any time via a web site.

#### Activity 3 – Page 234

1. Anke plays cricket. She has made a video to promote women's cricket. The file size of the video is 22 GiB.

Construct an expression to calculate how long it will take to transmit the video across a network that has a transmission speed of 54 Mbps.

$$(22 \times 1024 \times 1024 \times 1024 \times 8) / (54 \times 1000 \times 1000)$$

2. Erik uses the network to send a file to his colleague Jens. It takes 35 seconds for Jens to receive the file. The network transmits data at a rate of 40 Mbps.

Construct an expression to calculate the size of the file in mebibytes.

$$(35 \times 40 \times 1000 \times 1000) / (8 \times 1024 \times 1024)$$

3. Marty plays multi-user games online with other gamers. How might high latency internet connection affect her user experience?

Latency is the time between data being transmitted and when it arrives at its destination. High latency will mean gamers will experience a delay, or lag, when attempting to play a game, especially with multiple users.

## 4.1: Networks

### Activity 4 – Page 237

Wireless hotspots are available in public locations such as airports, cafés and hotels.

Write brief notes explaining how a wireless hotspot works and outlining the security issues associated with their hotspots.

Wireless hotspots allow connectivity to online services in public locations. Wireless transmission uses radio waves to create connections between devices, eliminating the need for wires.

Hotspots (and any wireless connection) can typically carry Wi-Fi signal up to 100 metres, at speeds of up to 10 Gps. This is ideal for access in public areas that contain multiple devices.

A wireless router manages communications on the network and has a built-in wireless access point enabling it to act as a gateway to the internet for all connected devices.

Wireless connection to a hotspot will usually require a key to gain access, although in some public areas this is not the case. A network that does not require a password to connect to it does come with security concerns. Because this kind of network has no encryption, there is the possibility that other people using the network could gain access to your internet usage.

All transmission can be intercepted, but without encryption, the data is clearly readable and understandable.

### Activity 5 – Page 239

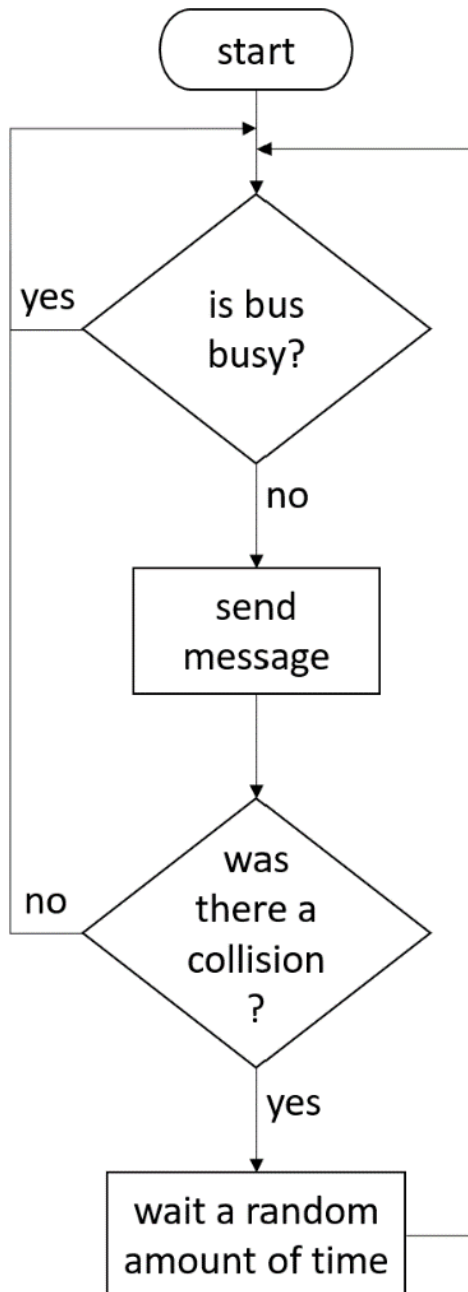
Copy and complete this table comparing different forms of wireless connectivity.

	Wi-Fi	Bluetooth	RFID	Zigbee	NFC
<b>Range</b>	100 metres	10 metres	450 metres	20 metres	< 20 cm
<b>Bandwidth</b>	3.2 Gbps	800 Kbps	0.25 MHz	20 Kbps	400 Kbps
<b>Energy Consumption</b>	Very low	Low to medium	Low, but increases the more readers collide	Very low	Very low
<b>Uses</b>	Portable devices	Headphones, speakers, wireless keyboard and mouse	Security tags, ID cards	Home smart hubs, thermostats	Smart cards, bank cards

## 4.1: Networks

### Activity 6 – Page 240

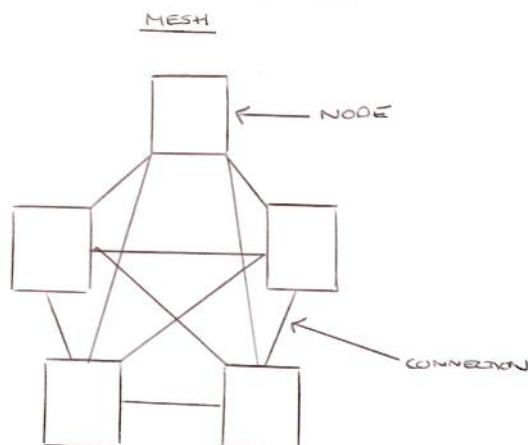
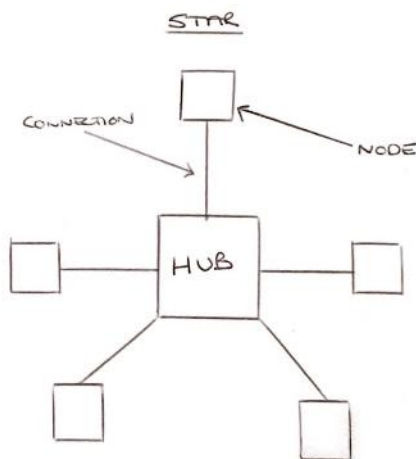
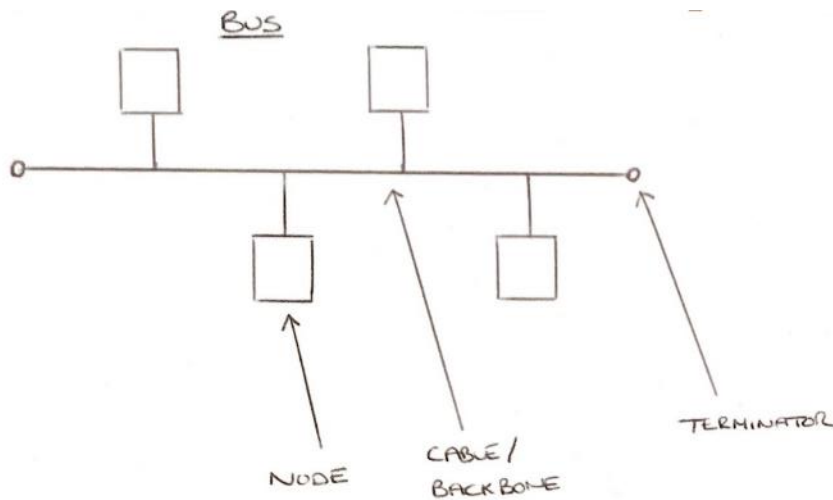
Draw the CSMA/CD algorithm as a flowchart.



## 4.1: Networks

### Activity 7 – Page 244

1. Close this book and draw annotated diagrams of the bus, star and mesh topologies.





## 4.1: Networks

2. Copy and complete this table comparing different network topologies.

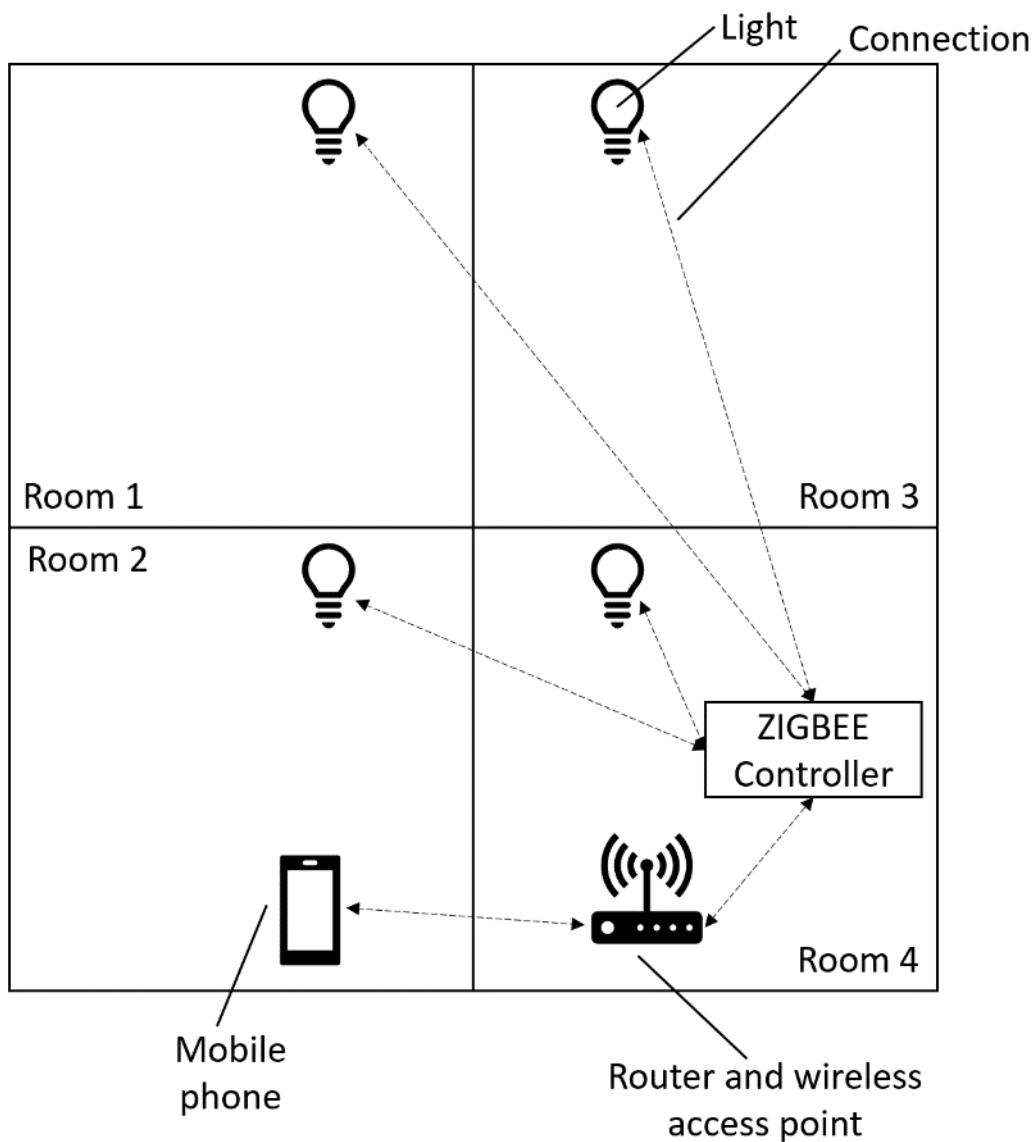
	Bus	Star	Mesh
<b>Characteristics</b>	All nodes connected to a single cable	Every node is connected to a central component. All data passes through this central component.	Each node is connected to multiple other nodes. Communication is peer to peer.
<b>Fault tolerance</b>	If one node fails, the network can still operate. If the cable is broken then all nodes fail.	If one node fails, the network can still operate. If the network is wired, a damaged cable will not cause any disruption to the rest of the network. If the central component fails, the entire network fails.	Very fault tolerant: if one node fails the network can still operate as another route can be found.
<b>Security</b>	All nodes on the network 'see' all data traffic. This could be a security risk.	Data traffic is only sent to the intended recipient. This makes it more secure.	In a fully connected mesh data can move directly between source and destination. In a partially connected mesh, data may be seen by in between nodes.
<b>Scalability</b>	The more devices added, the slower the network.	Nodes can be added and removed without taking the network offline.	Nodes can be added and removed without taking the network offline. Any number of nodes can be added.

## 4.1: Networks

<b>Cost</b>	Relatively low	Expensive to set up, as a lot of cabling is required.	Difficult and expensive to set up if wired
-------------	----------------	---	--

3. Layla wants to use a ZigBee mesh network to control the lights in her four-room apartment. She will use her mobile phone to switch the lights on and off remotely in each room.

Sketch a design for the lighting system.



### 4.1: Networks

#### Activity 8 – Page 246

The router on a hotel's network has been assigned a static IP address. Guests can log into the network using their own devices, which are assigned dynamic IP addresses.

- Why do you think the hotel has this set-up?

A dynamic IP address allows multiple devices to share the limited number of IP addresses available on a network for the hotel. This is useful because the hotels would have a maximum number of guests at any one time, so the IP addresses can be reused for different guests. The hotel connection for staff will not need to be changed, so can remain static.

- The Dynamic Host Configuration Protocol (DHCP) is used to assign dynamic IP addresses. Research DHCP and write a brief description of what it does.

DHCP stands for dynamic host configuration protocol and is a network protocol used on IP networks where a DHCP server automatically assigns an IP address and other information to each host on the network so they can communicate efficiently with other endpoints.

Further reading: go to [networkworld.com](http://networkworld.com) and search for their article 'DHCP defined and how it works'.

- How are static IP addresses assigned?

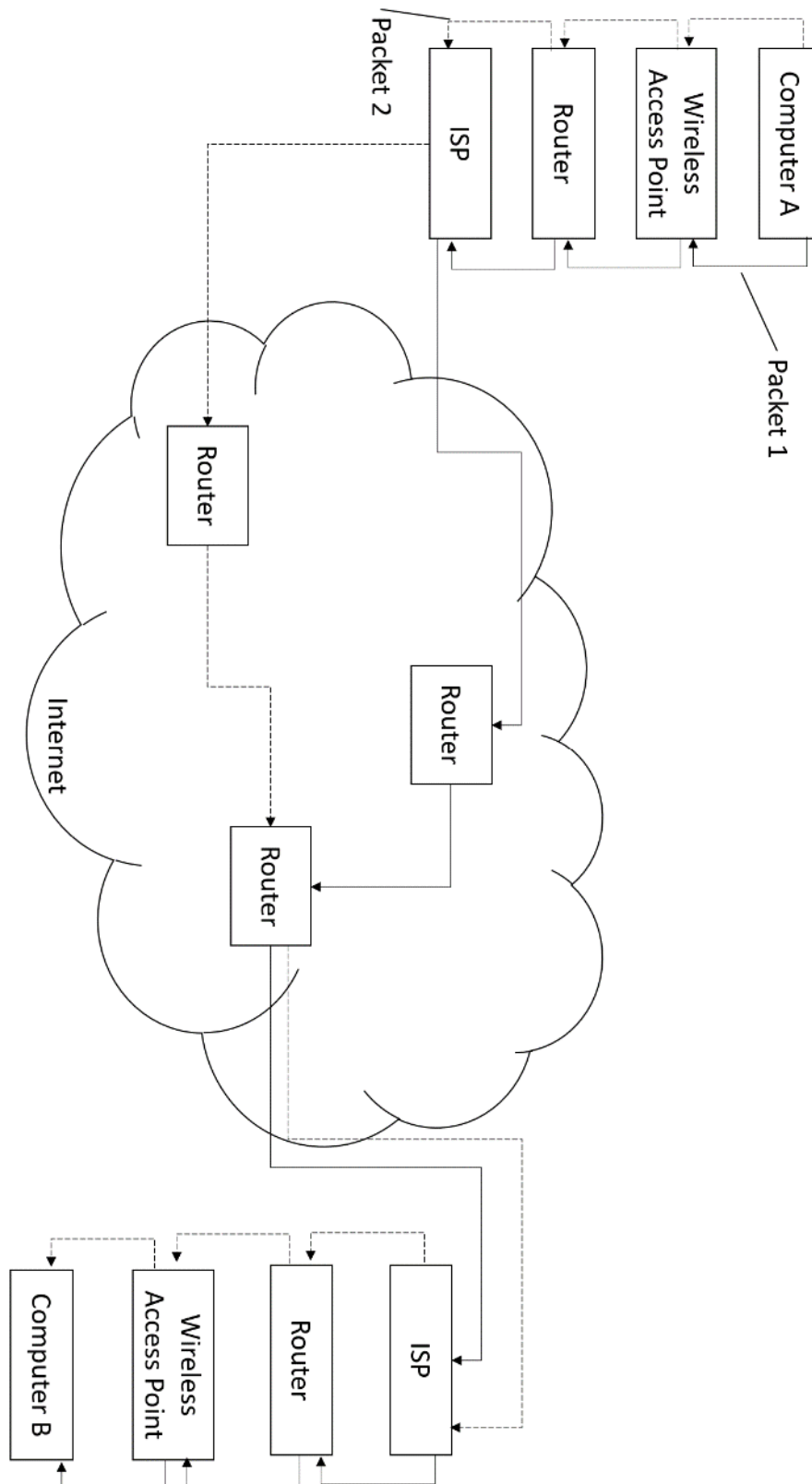
IP addresses are assigned to a host either dynamically as they join the network, or persistently by configuration of the host hardware or software. Persistent configuration is also known as using a static IP address.

Further reading: go to [wikipedia.org](http://wikipedia.org) and search for the entry on IP address.

## 4.1: Networks

### Activity 9 – Page 247

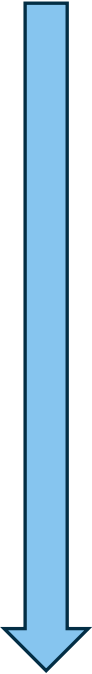
Draw a diagram to illustrate what happens when Computer A sends data to Computer B across the internet



## 4.1: Networks

## Activity 10 – Page 250

1. Close this book and draw an annotated diagram of the TCP/IP protocol layers – indicating what each layer does and what individual protocols run at each layer.

	Layer	Description	Protocols
	Application layer	Defines how various user services operate such as web browsers and email	HTTP, HTTPS, FTP, SMTP, POP3, IMAP
	Transport layer	Establishes a dedicated channel between the source and destination devices to transfer the data along	TCP
	Internet layer	Adds the source and destination IP addresses to the data packets received from the transport layer and routes them to the recipient	IP
	Link layer	Defines how data is transferred from one device to another along the whole route from sender to recipient	Ethernet, Wi-Fi

2. POP and IMAP are both email protocols operating at the Applications Layer of the TCP/IP stack. Summarise the difference between the two.

POP is a simple protocol that only allows downloading messages from your inbox to your local computer, whereas IMAP is much more advanced and allows the user to see all the folders on the mail server.

In POP3 the mail can only be accessed from a single device at a time, whereas IMAP messages can be accessed across multiple devices.

In POP3 all the messages downloaded at once, whereas the message header can be viewed prior to downloading in IMAP.

In older versions of POP3, the messages were deleted from the server after downloading. Now, POP3 is usually used to retrieve messages from an IMAP server, so the messages aren't deleted.

**4.1: Networks**

3. Alex uses a mobile phone and a laptop to access her email. Which type of email service would best suit her needs and why?

Alex should use IMAP because it allows messages to be accessed across devices as long as there is an Internet connection.

## 4.1: Networks

### Exam-style questions – Page 250

1. A supermarket wants to share data between head office and its 400 branches.

Explain why the supermarket would use a WAN rather than a LAN for this purpose.

(2 marks)

WAN is a wide area network meaning that data can be shared across vast distances. Therefore, 400 branches across a country would need to be connected by a wide area network rather than a LAN, which only covers a local area, for example, an office building.

2. Amin works for a marketing company.

Identify the type of network he uses for each of these activities.

(4 marks)

Situation	LAN	WAN
To review marketing materials stored on a server in the basement of the building	✓	
To work collaboratively with colleagues in the USA		✓
To print a leaflet on the printer located on the floor above Amin's office	✓	
To send an email to clients in Berlin		✓

3. A restaurant is considering whether to install a wired or wireless network.

Explain **two** benefits of each type of network.

(4 marks)

Wired networks give faster connection speed and decrease the amount of interference on the network connected.

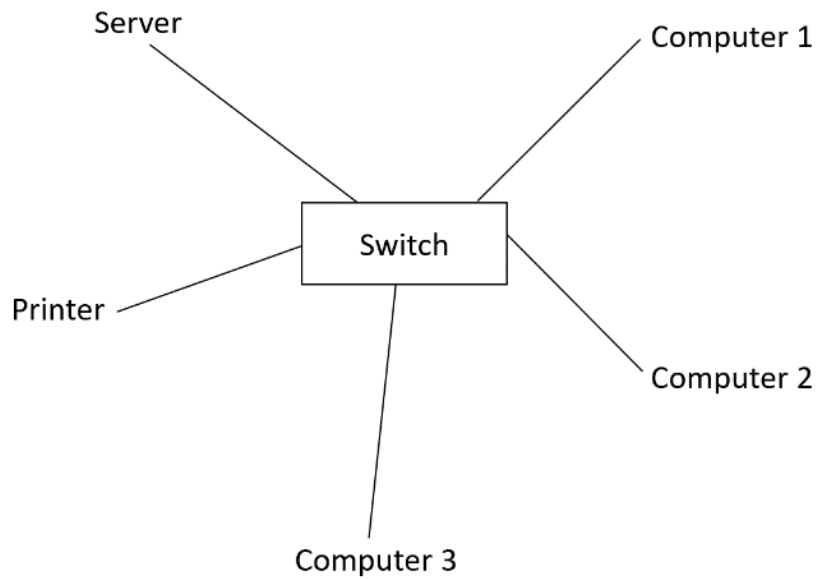
Wireless networks allow greater flexibility in adding devices to the network and are cheaper to install.

## 4.1: Networks

4. A small business has a network consisting of three computers, one printer and a server arranged in a star topology.

Draw and label a diagram showing this network.

(4 marks)





## 4.1: Networks

5. Complete the table to match each characteristic to a network topology. (3 marks)

Characteristic	Bus	Star	Mesh
Network performance degrades as more devices are added	✓		
All devices are directly connected to all others			✓
Each node has a physical attachment to a routing device		✓	

6. State the purpose of a network protocol. (1 mark)

Network protocols are sets of rules that dictate how to format, transmit and receive data between devices.

7. Describe how a router directs data traffic on the internet. (2 marks)

Routers forward data from one network to another across the internet from source to destination. They keep each other up to date about traffic conditions in their part of the network, so that data can be routed away from congested areas.

8. State which of the following protocols would not be used when transferring a web page from a webserver using the internet. (1 mark)

- a) TCP
- b) IP
- c) IMAP
- d) HTTP

9. A browser sends a request to a web server.

State the name of the TCP/IP layer that first handles this request. (1 mark)

Application layer

10. Two fields in a data packet transmitted across a network are source and destination.

Give two additional fields found in the header. (2 marks)

Sequence number and number of packets sent

### 4.2: Network security

#### Activity 11 – Page 253

Think about the data stored on your school network and try to answer these questions:

- Which bits of it are confidential?

Address and medical information

- How might this confidential data be of interest to a hacker?

It could be used to acquire personal information that could be sold.

- What would happen if the integrity of this data was compromised?

If data was stored incorrectly, it could lead to miscommunication with parents and other stakeholders.

- How might this occur?

Data could be incorrectly entered or edited by unauthorised employees, if suitable network privileges were not in place.

- Could the school continue to function if the network was unavailable?

Yes, but with a great deal of disruption. It would only be sustainable for a short period of time.

- Are there plans in place to deal with such a situation?

Separate servers would be available for different activities: for example, a file store is kept on a separate server to a printer server. On a more basic level, things like paper registers would be available as a contingency.

#### Activity 12 – Page 255

Passwords are the most basic form of authentication – but they are often easy to crack.

- Find out what type of password can be cracked using brute force tactics.

Passwords that consist of only letters, especially if formed to make words found in the dictionary, or names.

Passwords that conform to strict rules, such as eight characters long, one capital, one digit, and one special character.

Passwords to use digits instead of common letters (3E, 00)

- Describe four features of a secure password.

Include numbers, special characters, non-English words, unrelated/random words, long passwords more than ten characters, something easy for a human to remember but difficult for a computer program to randomly generate

- Password managers help to improve password security. How is this achieved?

Password managers store your login information for all the websites you use and help you log into them automatically. They encrypt your password database with a master password – the master password is the only one you have to remember.

### 4.2: Network security

- Biometric authentication offers several advantages over passwords. Name three.
  - Cannot be replicated by another person
  - Allows faster log in
  - Does not require remembering multiple passwords

- Can you think of any ways of beating biometrics?

Use a putty, such as plasticine, to take a mould of somebody's fingerprint and use it to replicate the original. Sometimes, photographs can be used to fool face scanning software.

#### Activity 13 – Page 256

A company has offices on two sites. Each site has its own LAN. The two LANs are connected together using a leased line. Data travelling across either of the two sites is unencrypted, but any data routed from one site to the other is encrypted.

Why do you think the company has opted to do this?

Company employees would need to have a username and password to access the network. This means that only authorised personnel would have access to the data; therefore, encryption is less of a priority as a security measure where data is shared between employees within the same site. The risk of data being intercepted by unauthorised individuals or organisations is low. Encryption and decryption require computational power and adds time to each transmission. Time-critical applications may be affected.

However, data should be encrypted for any sharing that happens across the leased line because the company does not have the same level of control over it. Because this cable runs outside of the building, there is a risk that the data could be intercepted; therefore, it should be encrypted. Encrypting the data would mean that, even if it was intercepted, the data would not be understandable.

## 4.2: Network security

### Exam-style questions – Page 258

1. A company designs processor chips.

It stores details of chip designs on a network file server.

Describe **two** ways in which unauthorised access to the design files can be prevented. (4 marks)

Physical protection:

Unauthorised access to the server could be prevented by installing physical security measures, such as a locked server room. This would mean that the room could only be accessed by authorised employees, such as a network manager or administrator. An additional measure would be to have an electronic lock system installed, so that only authorised employees could access the room with a key card or fob. The advantage of this is that employee activity can be recorded and tracked and, in the event of a key card or fob being lost, the card can be deactivated to avoid a card being used without authorisation.

Software protection:

Installing firewalls means anyone outside the company should be prevented from accessing the network.

User authentication details (logins) would protect the designs from anyone in the building where the networked machines are.

2. A company uses a hardware firewall to protect its network and installs software firewalls on all the portable devices that are permitted to access the network.

- a) Explain how the hardware and software firewalls work together to enhance data security on the network. (3 marks)

A software firewall should be installed on each device on a network. This will mean that default rules can be applied, so that the device is protected from harmful content that may affect the network. This is especially important if a device is portable and taken off-site away from the LAN or WLAN.

A hardware firewall should be installed additionally on the premises, as they provide more flexibility in terms of the rules that can be applied and allow faster data transmission. They will block or allow connections from devices outside the network.

Maximum protection of the network is achieved when both hardware and software firewalls are used together.

- b) Devices on the network are automatically logged out if they are inactive for more than 5 minutes.

Explain how this helps protect the network. (2 marks)

If a device is left inactive for 5 minutes, there is the possibility that the device, such as a laptop, has been left unattended. An unattended, unlocked device is

### 4.2: Network security

vulnerable to unauthorised access to potentially sensitive data that could be edited, copied or deleted either by mistake or intentionally. The device locking or being logged out after a set time minimises this risk as a password or biometric reading would be needed in order to regain access.

3. Keeping the operating system and application software up to date helps to keep the network secure.

Explain why this is the case.

(2 marks)

Application software, such as an internet browser or VoIP service, should be kept up to date when new versions or patches are released. Out-of-date software may not have the latest security features to protect individuals from the newest form of virus or malware that are developed.

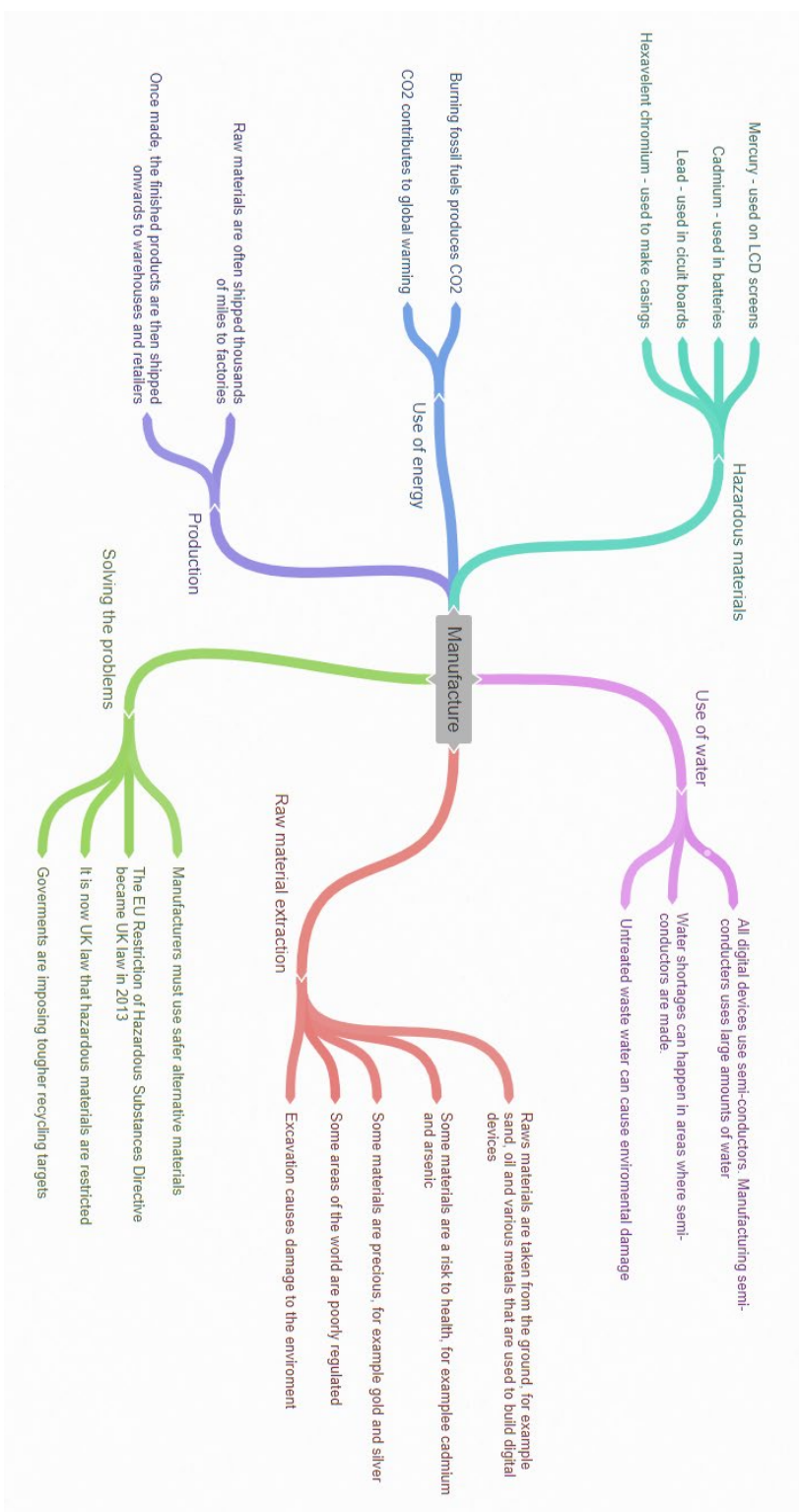
The application could have had vulnerabilities when it was released and they have only been spotted now. An update will correct those faults.

### 5.1: Environmental Issues

## Topic 5: Issues and impact

### Activity 1 – Page 262

Create a mindmap with 'Manufacture' at the centre. Complete the mindmap based on the paragraphs above. Include strands such as 'raw material extraction' and 'energy use'.



## 5.1: Environmental Issues

### Activity 2 – Page 262

Create a set of flashcards to help learn the terminology from this section.

The front of the card should contain the key term to define.

You can find these words highlighted in pink.

The back of the card should contain the definition. You can find the definitions in the glossary. However, you should write the definition in your own words. If there are words in the definition that you do not understand, then look them up in a dictionary or on a computing-related online dictionary. Make a definition that is really clear to you and that you can fully understand.

**Continue to add more flashcards as you progress through this section.**

Example flashcard:

**Replacement  
Cycle**

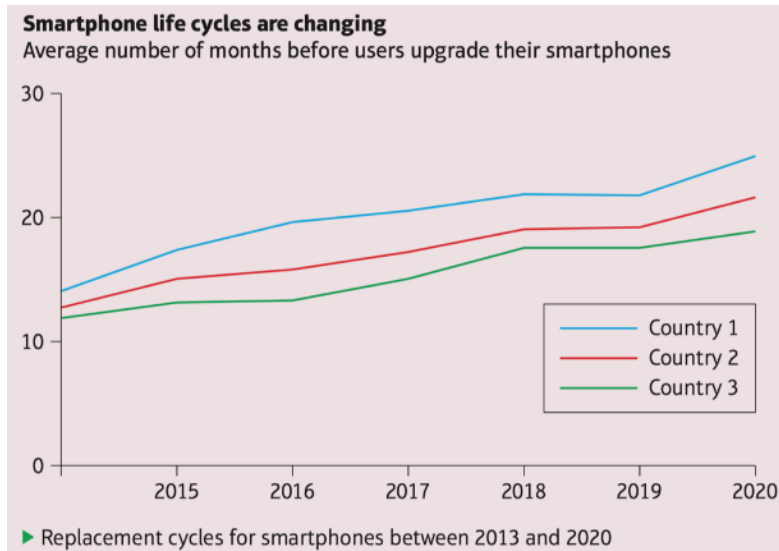
The period of time  
between the purchase of a  
product and  
its replacement with  
an equivalent product.

## 5.1: Environmental Issues

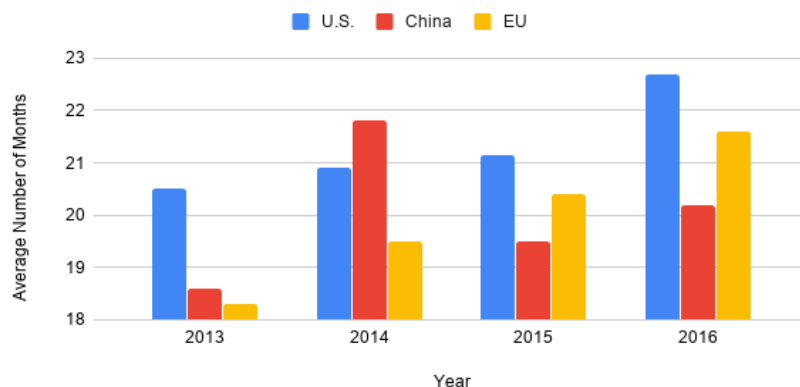
### Activity 3 – Page 263

Redraw this graph using a different vertical scale, e.g. from 18 to 23 so that the change in replacement cycles will be more apparent.

Describe the trend shown in the graph and give reasons for it.



Average number of months before users upgrade their smartphone



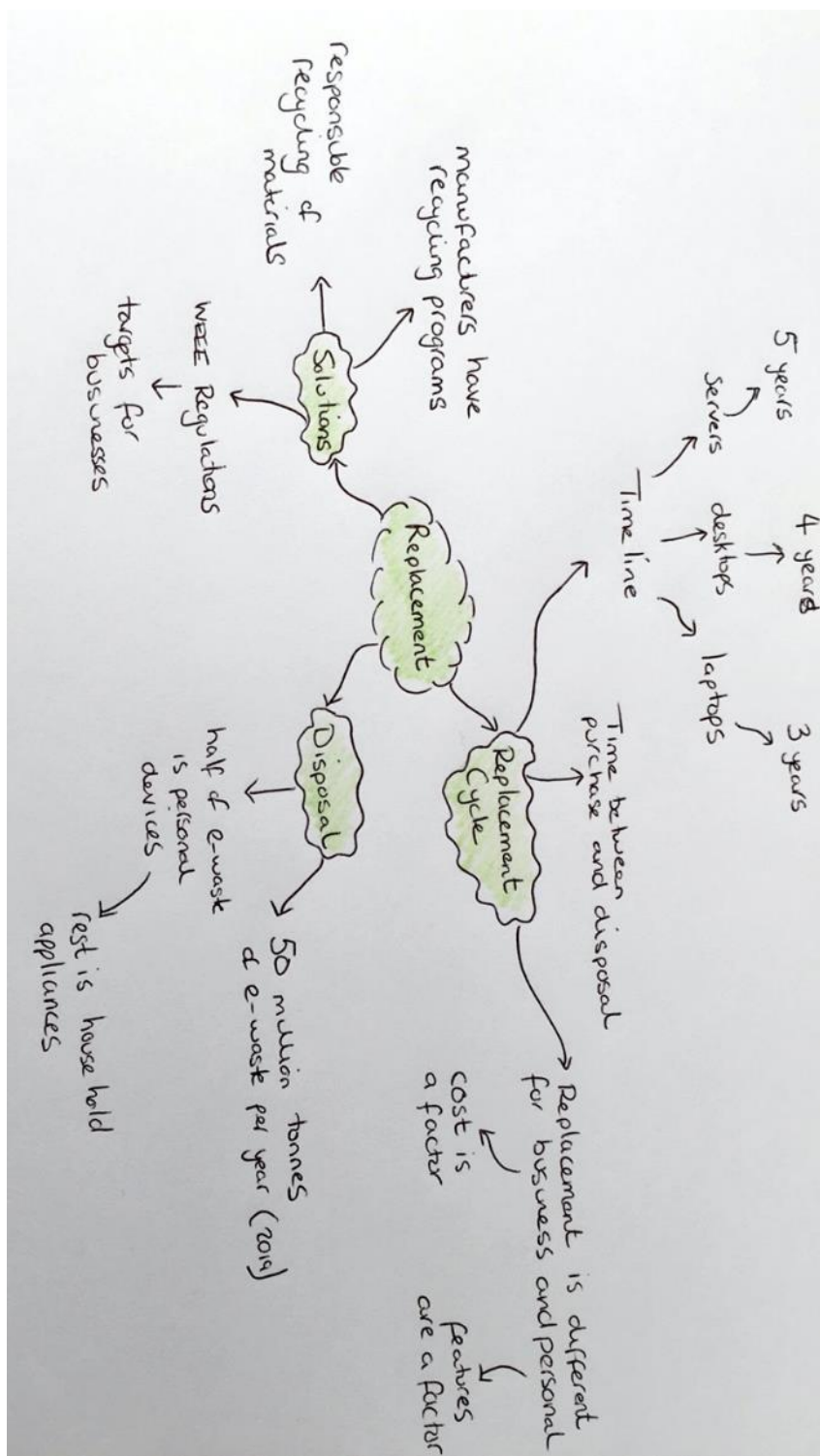
The trend in the graph shows that the average number of months before a user upgrades their device is increasing in both the US and the EU. China saw a spike in 2014, before dropping off in 2015, but is on the increase. This increase in months between upgrades could be because a new device upon release was too expensive for more users initially. Another reason could be that 2013 saw a large number of brand-new features being introduced, whereas subsequent releases only saw tweaks being made to the hardware. Additionally, the heightened awareness of recycling and sustainability is likely to have an impact on consumer-buying habits, meaning they are less likely to replace their devices at such a rapid pace.



## 5.1: Environmental Issues

### Activity 4 – Page 264

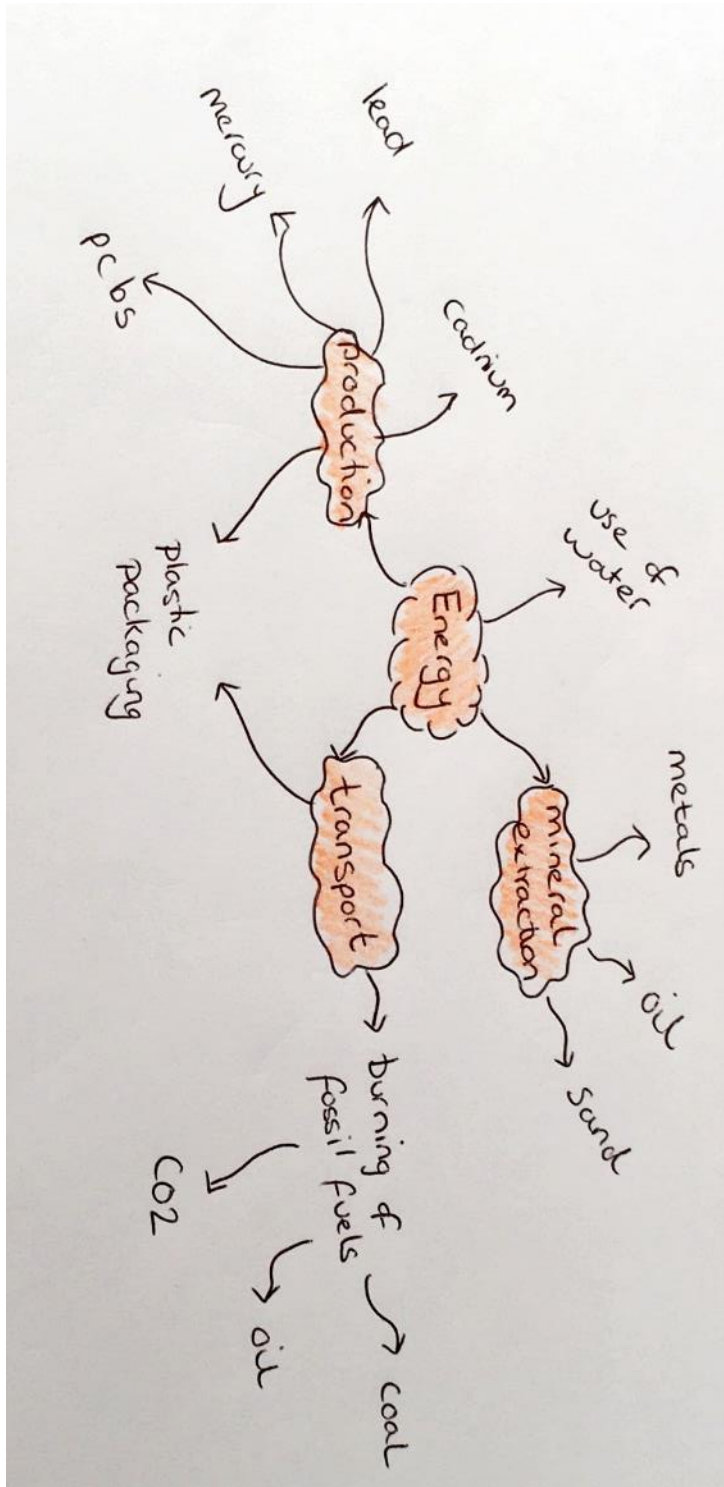
Create a mindmap with 'Replacement' at the centre. Complete the mindmap based on the paragraphs in the above section on disposal. Include strands such as 'replacement cycle', 'disposal', and 'solutions'.



## 5.1: Environmental Issues

### Activity 5 – Page 266

1. Create a mindmap showing how energy is consumed in all areas of digital technology. Have 'Energy' at the centre with links to area such as 'Mineral extraction', 'Transport', etc.



**5.1: Environmental Issues**

2. Revisit each section and identify the legislation put into place to limit the environmental effects of digital technologies. Identify each legislation, what it covers, and who enforces it.

Legislation	What is covers	Who enforces it
The EU Restriction of Hazardous Substances (RoHS) Directive	Restricts the use of hazardous materials in computing technology	The National Measurement Office
The Waste Electrical and Electronic Equipment (WEEE) Regulations (2013)	Sets targets for the collection, recycling and recovery of computing technology for business	Environment Agency

## 5.1: Environmental Issues

### Exam-style questions – Page 266

1. Describe **two** environmental issues associated with the manufacture of digital devices. (4 marks)

The production of digital devices has many environmental issues associated with it. The extraction of raw materials from the ground, which are used in the production of digital devices, can lead to intense damage to the local environment, including scarring of the landscape and contamination of ground soil and water supplies.

Once the raw materials have been extracted, the shipping of the materials also has an environmental impact. The transportation to manufacturers uses fossil fuels, such as oil and coal. This contributes further to global warming.

2. State **two** ways in which governments are attempting to control the environmental impact of digital device manufacture. (2 marks)

The EU Restriction of Hazardous Substances (RoHS) Directive became UK law in 2013.

Tougher recycling targets are set by governments to collect more reusable materials from computing technology when they are disposed of.

Recycling sites are being upgraded to take a wider range of materials, including appliances and smaller digital devices.

3. Explain how having a short replacement cycle for digital devices is harmful to the environment. (4 marks)

A short replacement cycle for digital devices is harmful for the environment because it results in an increase in demand, production and disposal over a shorter period of time. An increase in manufacture results in increases in extraction of raw materials, transportation of goods and energy usage in factories. In turn, these increase the burning of fossil fuels that contribute to global warming. Once a device is finished with, it is likely to end up at landfill, contributing to increasing amounts of e-waste already in the world.

4. Discuss the impact of computing technology on the environment. (6 marks)

There are several ways that computing technology harms the environment. The development of a device requires raw materials to be extracted from the ground. These include sand, oil and metals of various types. Precious metals such as silver, gold and copper are also extracted. This impacts on the environment because extraction of these materials can scar the landscape of the area and contaminate soil and local water supplies. Once the materials have been extracted, they need to be transported to factories for use in the building of a device. Both transportation of materials and manufacture of devices contribute to CO<sub>2</sub> emissions through the burning of fossil fuels. Additional transportation is then required to distribute the devices to suppliers and consumers. Once a device comes to the end of its replacement cycle, or when a consumer is done with the device, it is often disposed of in landfill if not recycled or reused. Increasing the length of the replacement cycle

## 5.1: Environmental Issues

of a device will contribute to the reduction of extraction, transportation, development of further disposal of devices and the subsequent impact on the environment.

5. Explain **one** reason why data centres use large amounts of energy. (2 marks)

Data centres house large servers that become very hot while operating. The servers need to be kept cool to stop them from malfunctioning and keeping them cool uses large amounts of energy to run air conditioners and or fans.

6. Give **four** measures that can be taken to make data centres more environmentally friendly. (4 marks)

- Use hot-aisle/cold-aisle to increase cooling system efficiency.
- Invest in research to develop new alternatives to silicon-based storage.
- Ration internet usage.
- Replace hard disk drives with solid state drives.
- Move to a location with naturally lower temperatures.
- Use naturally occurring energy, i.e. wind, thermal

## **5.2: Ethical and Legal Issues**

### **Activity 6 – Page 268**

Create a set of flashcards to help learn the terminology from this section.

The front of the card should contain the key term to define. You can find these words highlighted in pink.

The back of the card should contain the definition. You can find the definitions in the glossary. However, you should write the definition in your own words. If there are words in the definition given in the glossary that you do not understand, then look them up in a dictionary or on a computing-related online dictionary. Make a definition that is really clear to you and that you can fully understand.

**Continue to add more flashcards as you progress through this section.**

[See Activity 2 \(Page 262\)](#)

### **Activity 7 – Page 269**

The right to privacy must be balanced against the needs of society. Describe two situations in which you believe an invasion of privacy is justified. Explain your reasoning.

[One instance where an invasion of privacy may be justified is in the pursuit of criminal activity. For example, requiring an ISP to surrender Internet usage logs or breaking the password on a mobile phone may be done when the police or other part of the justice system is charging or trying a criminal case. Bodies must be given permission to do this by the courts or existing legislation. It cannot just be done on a whim or vague suspicion.](#)

[Another instance where an invasion of privacy may be justified is where there is concern that a person may be a danger to themselves or someone else. There are often examples of this in the news involving cases of bullying, stalking, deception, mental health issues, eating disorders, or the promotion of violence. These may involve looking at the contents of online accounts or digital devices.](#)

### **Activity 8 – Page 273**

At best spam is annoying but it can also lead to scamming and fraud. Originally rules-based spam filters looked for key terms like 'online pharmacy' or 'Bitcoin' in emails from unknown addresses, but spammers quickly outwitted these filters.

Carry out research and produce a poster showing how artificial intelligence and machine learning are being used by Google so that they can claim that Gmail successfully filters 99 per cent of spam.

Go to [cloud.google.com](https://cloud.google.com) and search for the article 'Spam does not bring us joy—ridding Gmail of 100 million more spam messages with TensorFlow'. It will be helpful in completing this task.

### **Activity 9 – Page 276**

Consider the production and use of lethal autonomous weapons from practical, ethical and legal points of view.

### 5.2: Ethical and Legal Issues

Possible discussion points could be the reduction in human fatalities, the disparity between countries with autonomous weapons and those without, and removing the 'humanity' of conflict with autonomous weapons.

Consider the use of facial recognition systems from practical, ethical and legal points of view.

Possible discussion points could include the potential for every aspect of all citizens' public life available as a digital recording; consideration for the required infrastructure, the energy to run that infrastructure, and the storage required for such a vast quantity of data; the requirements for new laws to determine how/when the data is to be used.

Are they accurate enough? Are they biased? Do they violate a person's right to privacy? Is there a legal and regulatory framework?

Possible discussion points could be algorithmic bias and ethnicity, how long and for what purpose recorded information is stored, where this data is stored and how it is made secure. For a useful article, go to [wired.co.uk](http://wired.co.uk) and search for 'Facial recognition is in London. So how should we regulate it?'

#### Activity 10 – Page 277

Carry out research to find three other employment sectors where the introduction of AI could lead to unemployment. Create a presentation to display your findings.

Sectors to research could include telemarketing, bookkeeping, receptionists, couriers, market researchers and retail workers.

#### Activity 11 – Page 277

In groups, discuss the ethical and legal implications of creating and using actual living robots. At present they are tiny but in future may be larger and acquire intelligence. Summarise the outcomes and produce a presentation for the rest of the class.

Possible discussion points could include sentience, human employment and human interaction.

#### Activity 12 – Page 278

Have you ever listened to some music or a song that reminded you of another one? The music industry is one area where there are lots of claims of copyright infringement. Carry out research and create a report of one such case, explaining the cause of complaint, the defence and the verdict.

Examples of cases could include Robin Thicke vs. Marvin Gaye (2014) or Lana Del Rey vs. Radiohead (2018).

## 5.2: Ethical and Legal Issues

### Activity 13 – Page 279

Create a table summarising the differences between a copyright, a patent and a licence.

Note: This question should be about the difference between a copyright, a patent, and a trademark.

Copyright	Patent	Trademark	Licence
Protects original works (books, music, art, etc.) or the expression of an idea (software)	Protects inventions	Protects words, phrases or design attached to a business	Grants permission to use software, but not own it
Work must be original	Invention must be new	A mark must be distinctive	Sets out what use is allowed (installation, copying, etc.)
Protects for life of creator plus 70 years	Protects for 20 years	Protects for as long as the mark is in use	Length will be set out in the licence, may be time limited or perpetual

### Activity 14 – Page 279

Create a table that summarises the differences between open-source and proprietary software.

Open-source software	Proprietary software
Free to use	Not free to use (unless freeware)
A community of people work together to update the software to make improvements	The developer of the product has complete control over updates
The user can make any changes they wish: copy, edit and share, as long as there is no fee charged	The user does not have access to the source code and is legally not allowed to change the software

### Activity 15 – Page 279

Review your flashcards. Are they complete? Do they cover all of the key words and concepts? Are the definitions accurate? [See Activity 2 \(Page 262\)](#)



## 5.2: Ethical and Legal Issues

### Exam-style question – Page 280

1. Explain one ethical concern associated with the use of surveillance cameras.

(2 marks)

Surveillance cameras can use face recognition to identify individuals. There is an ethical concern in this as personal data is being collected without permission of those being captured.

2. People often give their consent to their personal data being collected without realising they are giving it. Explain what the Data Protection Act (2018) expects organisations should do to ensure that consent is actually given. (4 marks)

The Data Protection Act (2018) expects organisations to allow users to say no to their data being used. The organisation must be specific about what data processing activities are being carried out. In addition, the user must know the identity of the organisation and have the option to withdraw their consent at any time.

3. Identify which of the three crimes in the Computer Misuse Act is being committed in the following examples and state the reasons for your decisions:

- a) A student sees that their teacher has left their computer unattended and alters their marks recorded by the teacher.

Intentional and unauthorised destruction of software or data

The student used the account of another person to maliciously change the data.

- b) A user accesses a computer to find out a person's credit card number and security code so they can use it to buy goods online.

Unauthorised access with intent to commit further offences

Whether the person had permission to use the computer or not, the intent is to commit a crime.

- c) A student tries to guess the password of a class member and tries to log into the network as that person. (6 marks)

Unauthorised access to computer material

The intent, in this case, even if harmless, is still a violation of the law.

4. State what is meant by 'artificial intelligence'. (1 mark)

Artificial intelligence is the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings.

5. Describe what is meant by 'algorithmic bias'. (2 marks)

Algorithmic bias are the tendencies of computer systems that create unfair outcomes. One example of this is a program that always chooses shortlist males over females for a job.

6. Explain how algorithmic bias can be introduced. (2 marks)

## 5.2: Ethical and Legal Issues

Algorithmic bias can be introduced into AI through pre-existing bias in data or by the programmers themselves. Algorithmic bias is usually unintentional.

7. Describe **one** ethical concern about the use of driverless vehicles. (2 marks)

A driverless vehicle may not make the same decision that a human driver would make in that same instance. They lack the capacity to make moral decisions in the event of a traffic collision incident, for example, being able to make a decision to avoid a car with a child in the back seat or a dog in a cage in the back of a different car.

8. Discuss the ethical and legal issues associated with the use of driverless cars. (6 marks)

The use of driverless cars brings with it some ethical and legal issues. Human drivers are often faced with difficult, split-second decisions that can affect the safety of passengers and other road users. Now these decisions will have to be made by an algorithm. In the event of a collision there are questions, both ethical and legal, that need to be considered. For example, who will be accountable? There is an argument for the non-driving owner, the manufacturer, the people who wrote the code for the car, etc. A self-driving car uses machine learning to make decisions: these would have been influenced by algorithms that were written by the programmers. This raises questions of cultural bias. One country will have their own acceptable rules that would be different to another country. A car designed and built in one country would potentially follow a different set of rules to what is acceptable if driven in another country. At present there is no agreement on any of these ethical rules, except that they should not be made by the manufacturers of driverless vehicles.

9. A software engineer produced a program to solve a particular problem and copyrighted the code. A few months later, a rival brought out another program that solved the problem in a different way.

Had the rival infringed the original engineer's copyright? Explain your answer.

(2 marks)

The rival engineer has not infringed on the original copyright because copyright only protects the expression of an idea, not the idea itself. The original source code is protected, but there's nothing to stop someone else from copying the idea and writing a program that essentially performs the same task.

## 5.3: Cybersecurity

### Activity 16 – Page 282

There are many similarities between a virus, a worm and a trojan, but there are some differences. Complete the table to show the characteristics common to all three and the distinguishing characteristics of each. You will need to add more rows.

Common	Virus	Worm	Trojan
Malicious and harmful effects	Hidden within other programs or files	Programs in their own right (do not exist inside other programs or files)	Programs in their own right (do not exist inside other programs or files)
Disrupt the functioning of a computer system	Designed to replicate	Do replicate	Do not replicate
Gain unauthorised access to a computer system	Copies inserted into other programs or files	Do not attach to other files	Do not attach to other files
Gather information from the users without their knowledge	Passed to other computers by users themselves	Distribute themselves independently of a user via email or network connection	Must be installed by user
		Can create back doors	Can create back doors
	Corrupts or deletes data on a disk	Corrupt or damage systems	May delete files and system information

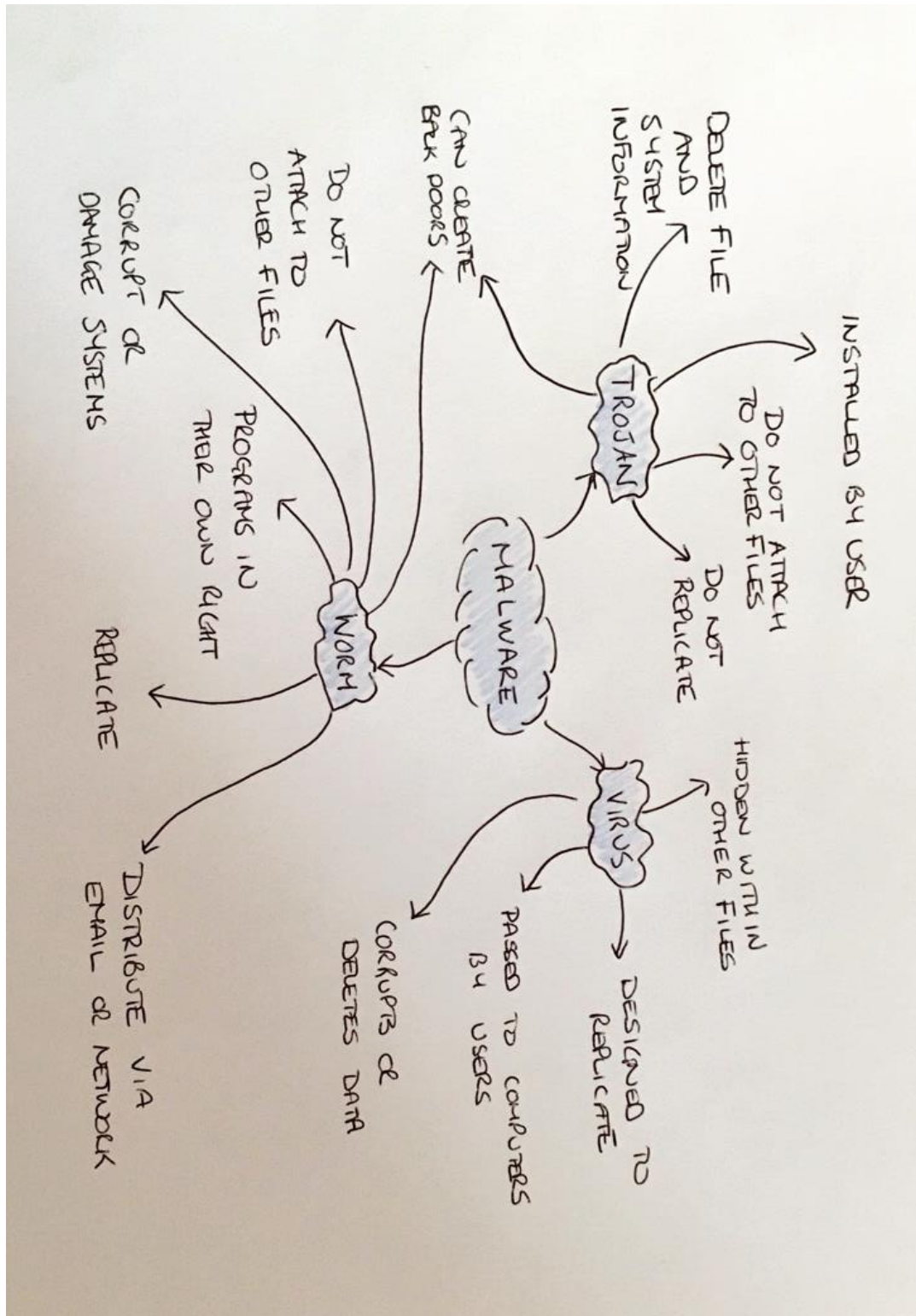
Similarities and differences between some types of malware.

## 5.3: Cybersecurity

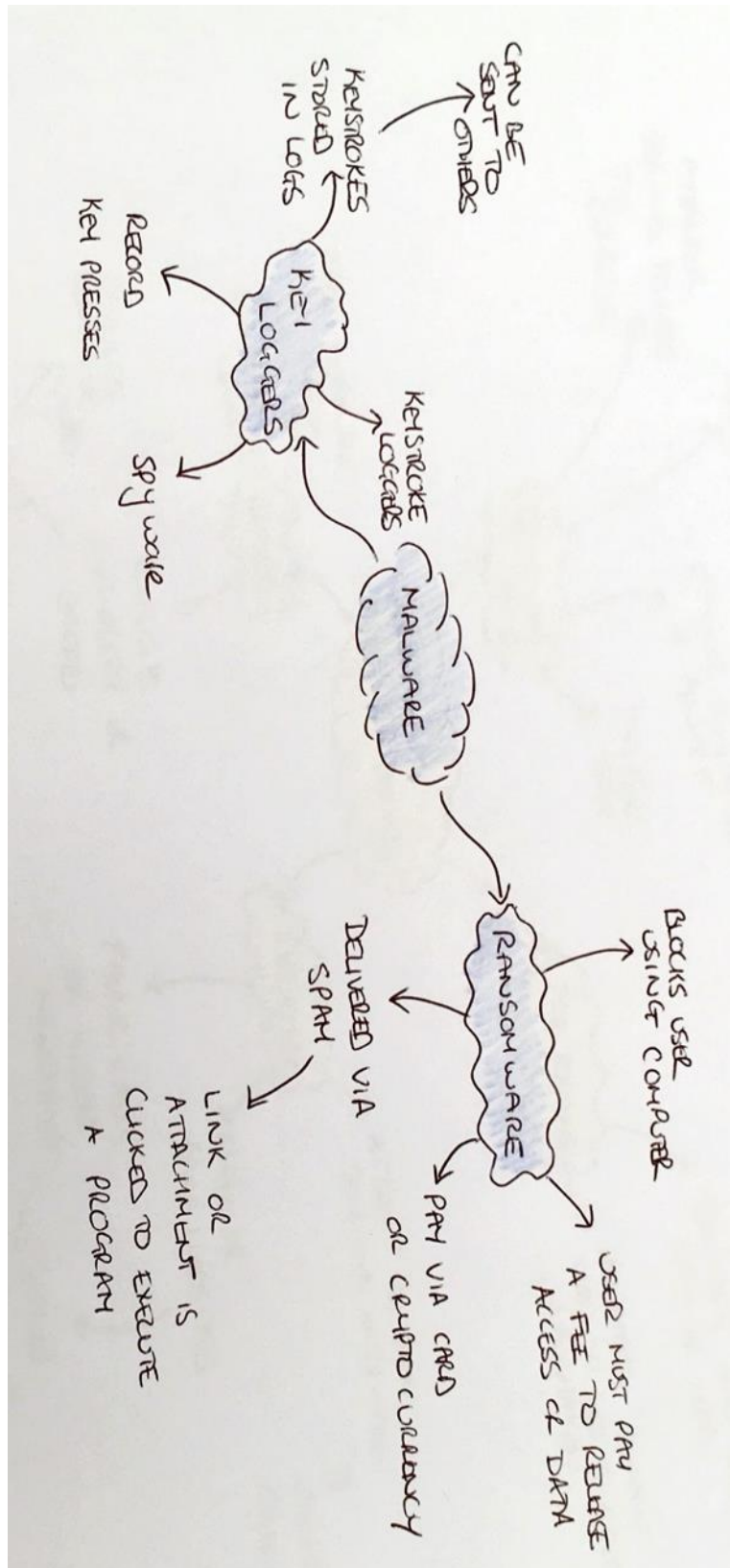
### Activity 17 – Page 284

Create a mindmap with 'Malware' in the centre and with links to different types of malware.

Note: There are two mindmaps here, but you may only have one. Be sure to cover all the different types of malware.



## 5.3: Cybersecurity



### 5.3: Cybersecurity

#### Activity 18 – Page 284

Investigate the ransomware attack on Travelex in early January 2020.

Write a summary of the attack. Consider these aspects:

- What did the attack look like to Travelex?
- What individuals or organisations were affected?
- Who were responsible?
- Why did they believe Travelex would pay?
- How did Travelex respond?
- Was it an appropriate response? Why?
- What was the impact of the attack on individuals, organisations and Travelex?
- What could Travelex do in the future to avoid the same kind of attack?

Lots of information online for this question, some good places to start students' research are:

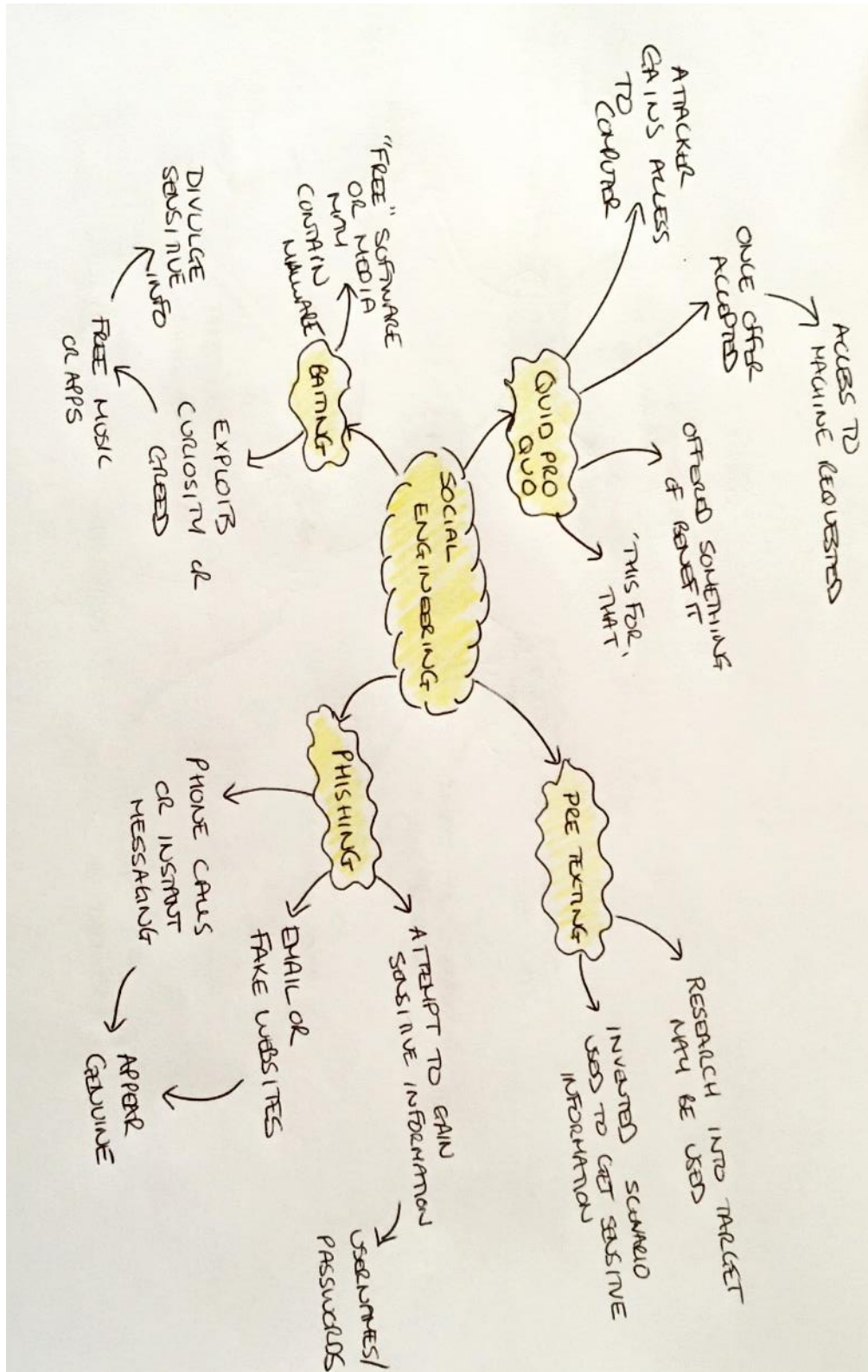
- [bbc.co.uk](https://www.bbc.co.uk) – search for the article 'Travelex being held to ransom by hackers'.
- [infosecurity-magazine.com](https://www.infosecurity-magazine.com) – search for the article 'Why the Travelex Incident Portends the Changing Nature of Ransomware'.
- [computerweekly.com](https://www.computerweekly.com) – search for 'Cyber gangsters demand payment from Travelex after 'Sodinokibi' attack'.



## 5.3: Cybersecurity

### Activity 19 – Page 286

Create a mindmap with 'Social engineering' at the centre and with links to different types of social engineering.



### 5.3: Cybersecurity

#### Activity 20 – Page 288

Create a poster providing information on the need for keeping software up to date and using anti-malware.

- Poster should include information on:
- keeping software up to date using patches to fix security issues
- considering replacing software that is no longer supported by the creator, which may be vulnerable in the future
- installing anti-malware software that either:
  - scans for malware using signatures
  - signature files are databases
  - signature files, themselves, must be kept up to date
  - detects malware using heuristics
  - should also be updated to ensure the use of latest knowledge about malware
- ensuring that anti-malware is scanning downloads from the Internet
- ensuring that anti-malware can, if required, scan portable devices attached to the computer

#### Activity 21 – Page 291

Investigate the acceptable use policy in use at your school or college. What are users allowed and not allowed to do? Is it the same policy for students and staff? How recently was it updated? Present your findings to the rest of the class.

There will be variations on this activity depending on the institution. However, a generic template can be found at [getsafeonline.org](https://getsafeonline.org) – search for ‘Sample Acceptable Usage Policy’. Your school or college policy is most likely to include elements of this template.



### 5.3: Cybersecurity

#### Exam-style questions – Page 292

1. Define what is meant by the term malware. (2 marks)

Malware is software that has been designed to gain unauthorised access to a computer system in order to disrupt its functioning or collect information without the user's knowledge.

2. Describe a situation where a key logger may not be considered malware. (2 marks)

Monitoring software used on a school network is not considered malware because the recorded keystrokes are not being used for malicious purposes.

3. State the name of the malware that does not need a user to distribute it. (1 mark)

Worms

4. State the name of the malware that embeds itself into other programs or files. (1 mark)

Viruses

5. Explain the connection between ransomware and cryptocurrency. (2 marks)

Ransomware is a type of malware that blocks users from using their computer or accessing the data stored on it, until a fee (ransom) has been paid to release it. The fee is then requested to be paid via cryptocurrency.

6. 'The weak point in any system is the human.' Describe two examples of cyberattacks that support this statement. (4 marks)

Two examples of cyberattack that require human intervention for them to be executed are trojans and phishing. Trojans are malicious software that must be installed by the user and are hidden within what appears to be authentic software, often sent via email. Phishing attacks are also often spread via what appear to be legitimate email communications, but instead are designed to extract sensitive or confidential information from the user.

7. Define what is meant by the term 'baiting cyberattack'. (2 marks)

An attack that exploits a person's curiosity or greed is known as baiting. The victim is enticed by the offer of something for free, such as a music download or an app upgrade. As part of the process for obtaining the freebie, the victim may divulge sensitive information, such as login details.

8. State two ways that anti-malware software may identify an infection. (2 marks)

Anti-malware software can be installed to prevent infection by malware and search for and destroy it if it has infected the system. One way that malware identifies infection is using a database of malware definitions. Another way anti-malware software detects malware is by a form of analysis called heuristics. Heuristics identifies malware by behaviours and characteristics, instead of comparing against a list of known malwares.

9. Explain why users should always apply patches to their software. (2 marks)

### 5.3: Cybersecurity

Patches should be applied to software because they correct errors in the program or vulnerabilities in the security of the software application.

10. Describe the role of a signature file in anti-malware software. (2 marks)

A signature file is used to identify known malware. Periodically the program will check online to see if there is a signature file newer than the one currently installed on a machine. If there is a new file, then it will be downloaded and updated.

11. An acceptable use policy states that employees must not attach one of their own portable devices to the network. State two reasons for this policy. (2 marks)

Employees should not attach their own portable devices to the network because the device may contain malware of some kind that could be transmitted to the network. Additionally, if the device were to contain malware of some kind it would jeopardise the security of the network, potentially harming other users and the network itself.

12. Discuss procedures that an organisation should include in its backup policy. (6 marks)

An organisation should include a variety of procedures in a backup policy, including procedures for recovery in the event of natural or man-made disasters. The policy should identify who is responsible for the backup process and which data is to be backed up, for example, whether all the data is to be backed up or an incremental backup. The policy should also include how often backups should be made (hourly, daily or weekly) if the data is changing on a continuous basis. The policy will need to identify the medium on which the backup is stored, for example, tape, DVD or cloud storage. Further policy guidelines would include the location of the backup and for how long they should be kept.

As part of the backup policy, a set of procedures should be in place for the recovery of data. This would include how often testing takes place, a list of all possible threats and how to counteract them and a statement of where all backups are kept. These should also include a list of personnel and who is responsible for which part of the recovery process, such as finding a new building, buying/installing machines, and retrieving and restoring software and data.

## **Prepare for your exam – Paper 1**

The answers for each of the sample questions in Paper 1 are provided in the textbook on pages 294-301.

## Prepare for your exam – Paper 2

### Question 01 – Page 304

#### Provided

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False          # Assume everything is invalid
5  userChoice = 0              # Set to an invalid value
6
7  # -----
8  # Subprograms
9  # -----
10
11 def showMenu():
12     print ("Option 1")
13     print ("Option 2")
14     print ("Option 3")
15
16 # -----
17 # Main program
18 # -----
19
20 # =====> Write our code here
21
22 print ("You entered option " + str(userChoice))

```

#### Response

```

1  # -----
2  # Global variables
3  # -----
4  validChoice = False          # Assume everything is invalid
5  userChoice = 0              # Set to an invalid value
6
7  # -----
8  # Subprograms
9  # -----
10 def showMenu():
11     print ("Option 1")
12     print ("Option 2")
13     print ("Option 3")
14
15 # -----

```

## Paper 2 – Application of Computational Thinking

```

16 # Main program
17 # -----
18
19 # =====> Write our code here
20 while (not validChoice):          # Keep user in loop
21     showMenu()                   # Call subprogram
22     userChoice = int (input ("Enter an option: "))
23
24     # Check if choice is valid
25     if (userChoice >= 1) and (userChoice <= 3):
26         validChoice = True
27     else:
28         print ("Invalid option, try again ")
29
30 print ("You entered option " + str(userChoice))

```

### Question 02 – Page 307

Provided

```

1 # -----
2 # Import Libraries
3 # -----
4 import math
5
6 # -----
7 # Constants
8 # -----
9 FILENAME = "MovieRatings.txt"      # Name of file
10
11 # -----
12 # Global variables
13 # -----
14
15 # -----
16 # Subprograms
17 # -----
18
19 # -----
20 # Main program
21 # -----

```

MovieRatings.txt data file

```

1 22,Two Lost Worlds,3
2 44,The Man in the White Suit,8

```

## Paper 2 – Application of Computational Thinking

```

3 66,Red Planet Mars,7
4 88,The Magnetic Monster,2
5 11,Creature from the Black Lagoon,8
6 33,Tarantula,5
7 55,Forbidden Planet,9
8 77,The Unearthly,9
9 99,The Blob,6
10 122,Journey to the Center of the Earth,7

```

### Response

```

1 # -----
2 # Import Libraries
3 # -----
4 import math
5
6 # -----
7 # Constants
8 # -----
9 FILENAME = "MovieRatings.txt"          # Name of file
10
11 # -----
12 # Global variables
13 # -----
14 file = []
15 average = 0.0
16 count = 0
17
18 # -----
19 # Subprograms
20 # -----
21 def loadFile():
22     file=open(FILENAME,"r")
23     text = file.readlines()#read all contents as array
24     data = []
25     for line in text:
26         lin = line.rstrip()#remove newline char
27         l = lin.split(',')
28         data.append(l)
29     return data
30
31 def calculateAverage(table):
32     total = 0
33     for record in table:
34         rating = int(record[2])#always round up

```

## Paper 2 – Application of Computational Thinking

```
35     total += rating
36     return (math.ceil (total/len(table)))
37
38 def countAboveAverage(average,table):
39     count = 0
40     for record in table:
41         rating = math.ceil(float(record[2]))#always round up
42         if rating > average:
43             count += 1
44     return count
45
46 # -----
47 # Main program
48 # -----
49
50 print("Loading data...")
51 file = loadFile()
52 average = calculateAverage(file)
53 count = countAboveAverage(average,file)
54
55 print("Average film rating:",average)
56 print("Number of films rated higher than average:",count)
```