1(a) A program asks a user to enter a number. Only integer values between 1 and 100 (inclusive) are accepted. The program is tested.

Tick (✓) **one or more** boxes in each row to identify the type or types of test for each example of test data.

Example test data	Normal test data	Boundary test data	Invalid / erroneous test data
27	✓		
"Hello"			✓
105		✓	
100	✓		

2

[4]

(b) An algorithm is written for the program.

i. Complete the algorithm for the program.

```
num = input("Enter a number between 1 and 100")

if num <1 . AND >100 then
    print("accepted")
else
    print("not accepted")
endif
```

[3]

ii. Give the identifier of **one** variable used in the algorithm in part **(i)**.

num

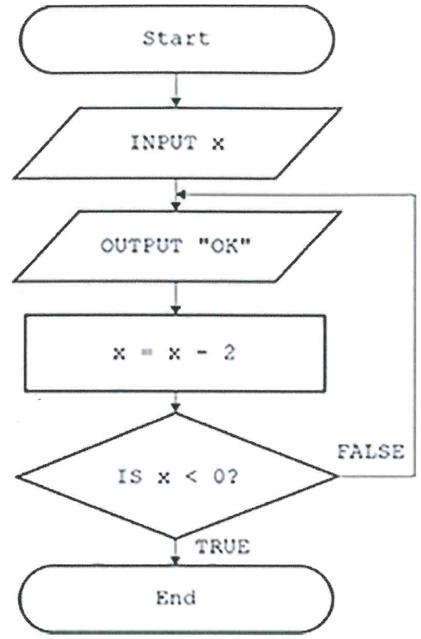
[1]

iii. Give **one** Boolean operator used in the algorithm in part **(i)**.

AND

[1]

2(a) This flowchart uses iteration.



Complete the table by identifying how many times "OK" will be output for when each value is input.

Input value	Number of times "OK" is output
3	2 ✓
10	8 ✓

2

[2]

(b) Iteration is a programming construct.

i. Identify **and** describe **two** different examples of iteration that can be used in a high-level language.

1 For loops which run for a pre determind ammount of ~~times~~ iterations

2 While loops which run until a condition is met.

[4]

4

ii. Give the names of **two** other programming constructs that are used in a high-level language.

1 Python

2 Pseudocode

[2]

0

(c) A programmer implements the flowchart as a program in a high-level language using an Integrated Development Environment (IDE).

Identify and describe **two** tools that the programmer could use in the IDE to create a program from the flowchart.

Tool 1 Debugger

Description Allows the programmer to find and fix errors

Tool 2 Runtime

Description Being able to run and test the ~~data~~ code.

[4]

4

3(a) Show the steps that a merge sort would take to put the data set into ascending numerical order.

2, 7, 3, 1, 9, 6, 5, 4

2,7 3,1 9,6 5,4

2,7,3,1 9,6,5,4

1,2,3,7 4,5,6,9

1,2 3,7 4,5 6,9

[4]

0

(b) Tick (✓) **one** box in each table to complete the descriptions.

An insertion sort…

	Tick (✓) one box
uses an array of numbers with a sorted part and unsorted part	
picks out the middle value to begin with	✓
merges arrays of numbers together	

A bubble sort compares pairs of values and then:

	Tick (✓) one box
always swaps them	
swaps them if they are in the correct order	
swaps them if they are in the incorrect order	✓

A sorting algorithm can sort data in an array. An array:

	Tick (✓) one box
is declared with a fixed length	
stores multiple values that cannot be changed while the program is running	✓
can only be declared with a maximum of one-dimension (1D)	

[3]

(c) An algorithm can be used to search for a value in an array. The algorithm follows these steps:

- compares the first value in the array to the value being searched for
- stops if the comparison is `true`
- moves to the next value in the array if the comparison if `false`.

i. Give the name of the searching algorithm described.

Binary search

[1]

ii. Identify **one other** condition that will cause the algorithm to stop.

If it finds the value

[1]

This was starting it instead for another.

4(a) An algorithm is written to read numbers from the external text file `data.txt`

Each number is checked when it is read from the external text file. The number is output to the user if it is not 0.

Complete the algorithm using the OCR Exam Reference Language statements in the box.

Not all statements are used.

`data`	`for`	`if`
`myFile.close()`	`myFile.readLine()`	`input()`
`myFile.writeLine()`	`open`	`print`
`temp`	`txt`	`x`

```
myFile = myFile.readLine()("data.txt")
while NOT myFile.endOfFile()
    temp = x
    if int (temp) ! = 0
        print..(temp)
    endif
endwhile myFile.close()
```

[5]

(b) Identify how casting is used in this algorithm.

To read the data of the file.

[1]

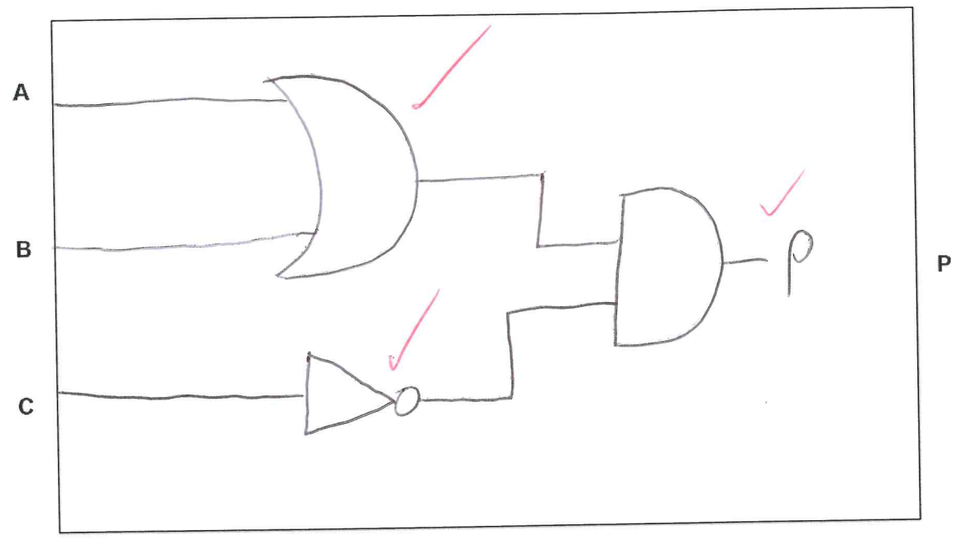
5(a) A security system protects online credit card payments. The system has three inputs:

Input	Description
A	`True` when the user enters their correct password
B	`True` when the user enters their correct PIN
C	`True` when the card is blocked

The output (`P`) is `True` if both of the following conditions are `True`:

- the user has entered either their correct password or has entered their correct PIN
- the card is **not** blocked.

Draw the logic circuit for the security system



[3]

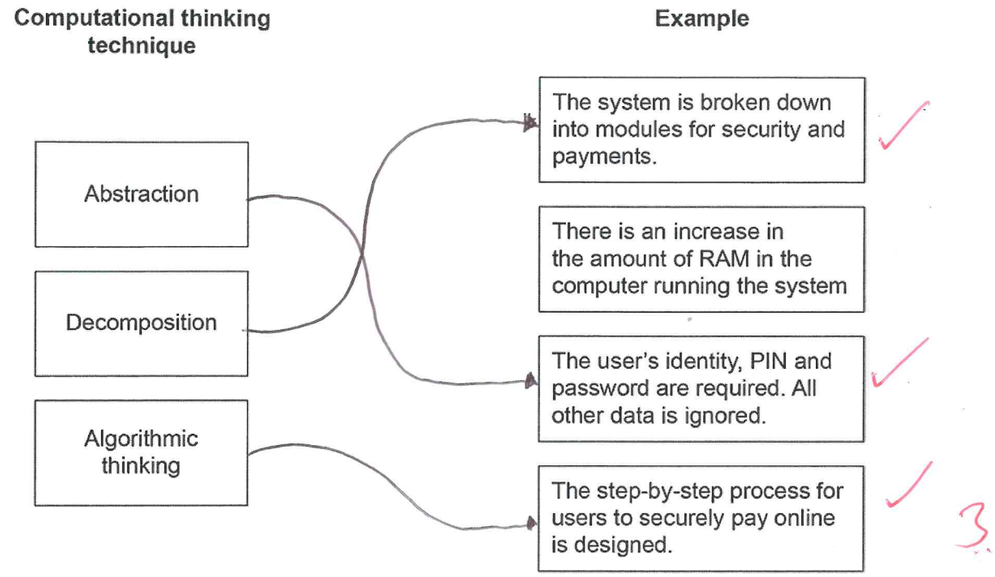
(b) Complete the truth table for **P = A AND B**

A	B	P
0	0	0
0	1	0
1	0	0
1	1	1

[2]

(c) Computational thinking techniques were used to design a security system.

Draw **one** line to match each computational thinking technique to its correct example.



[3]

(d) As part of a security system, the user is asked to choose a 4-digit value for their PIN. This PIN cannot be 1234 or 4321

Create an algorithm that:

- takes the PIN as input as a string
- checks that the PIN is 4 characters in length
- checks that the PIN is **not** either "1234" or "4321"
- outputs "`VALID PIN`" if the PIN meets all of the criteria
- outputs "`INVALID PIN`" if the PIN does not meet all of the criteria.

You do **not** need to check that the digits entered are numeric.

```
PIN = str.input("choose a 4 digit PIN")
if PIN == "1234" OR PIN == "4321"
    print ("INVALID PIN")
else
    print ("VALID PIN")
```

2

[6]

6(a) OCR Rockfest is a yearly music festival. Visitors buy tickets to attend and watch the music acts.

A computer system is created to sell tickets and store information about visitors and the music acts.

Visitors are asked to choose a password. The password is entered twice and stored in the variables `passA` and `passB`

i. Tick (✓) **one** box to identify the correct OCR Exam Reference Language statements for the design of this task.

OCR Exam Reference Language to enter two passwords	Tick (✓) one box
`passA and passB = input("enter password twice")`	
`input("enter your password") = passA` `input("enter your password again") = passB`	
`passA = input("enter your password")` `passB = input("enter your password again")`	✓
`"passA" = input("enter your password")` `"passB" = input("enter your password again")`	

[1]

ii. Design an algorithm that:

- checks if the passwords stored in `passA` and `passB` are the same
- outputs "`passwords match`" if they are the same
- outputs "`passwords do not match`" if they are **not** the same.

~~while~~ if

Sadly it needed an if not while.

```
passA == passB
    print("passwords match")
else
    print("passwords do not match")
endwhile
```

2

[3]

(b) The database table `TblMusic` stores data about the music acts.

Some of the data in `TblMusic` is shown:

ActID	Stage	Day	SetLength
1	Platinum	Saturday	0.5
2	Hills	Saturday	1.5
3	Triangle	Sunday	1.0
4	Platinum	Sunday	0.5

An SQL statement has been written to show `ActID` and `Stage` for all acts playing on Saturday. The SQL statement is incorrect.

```
SELECT ActID AND Stage
FROM TblActs
IF Day = Saturday
```

i. Refine the SQL statement to correct the errors.

SELECT ActID ~~AND~~ Stage
FROM TblMusic
~~IF~~ WHERE Day = Saturday

Has to match the table capital M.

[4]

ii. Identify one example of a record from `TblMusic`

1, platinum, saturday, 0.5

[1]

iii. Give the most appropriate data type for each field shown.

Field name	Data stored	Data type
Stage	The name of the stage	~~integer~~ String
SetLength	The length of each set in hours	integer

[2]

(c) Tickets to the festival are £60 each. Visitors can buy multiple tickets. For example, a visitor buying 4 tickets would pay £240.

i. Write a function, `calculatePrice()` that:

- takes the number of tickets to be purchased as a parameter
- calculates and returns the total price to pay.

You must use either:

- OCR Exam Reference Language, or
- a high-level programming language that you have studied.

This needs def to define

def calculateprice() — No parameter
x = ~~str~~ int(input ("How many tickets would you like to buy?"))
~~input = x * 60~~
Price = x * 60
print (Price)

[4]

ii. Write an algorithm that:

- uses the function `calculatePrice()` to find the price for a visitor buying 3 tickets
- stores the result in the variable `totalPrice`

You must use either:

- OCR Exam Reference Language, or
- a high-level programming language that you have studied.

~~totalprice()~~
calculateprice()
print (totalprice)

[2]

(d) The number of different music acts on each of the three stages is stored in the array `actNumbers`.

Each index of the array represents one stage.

Index	0	1	2
Data	10	14	6

`actNumbers`

This algorithm adds up the total number of acts on all three stages.

```
01 count = 0
02 for x = 0 to 2
03 count = count + actNumbers[x]
04 next x
05 print(count)
```

i. Complete the trace table for the algorithm. You may not need to use all rows in the table.

Line number	count	x	Output
1	0	0	
2	10		
3	10	1	
4	24		
5	24	2	
6	30		30
7			
8			
9			
10			

[4]

ii. The array `actNumbers` is changed to include data for two more stages. The new array is shown:

Index	0	1	2	3	4
Data	10	14	6	9	11

`actNumbers`

The algorithm from part (i) needs refining to work for the new array. The algorithm is repeated here:

```
01 count = 0
02 for x = 0 to 2
03 count = count + actNumbers[x]
04 next x
05 print(count)
```

Identify the line number that will need to be changed for the new array and write the updated line of code.

Line number 02

Updated code for x = 0 to 4

[2]

iii. Refine line 05 in the algorithm to output `"The total act count is "` and then the content of the variable `count`

For example:

```
The total act count is 50
```

New line 05 print("The total act count is " , count)

[1]

(e) There are 500 tickets available for the music festival. The number of tickets currently available is stored in the variable `tickets`

Complete the algorithm to:

- prompt the user to enter how many tickets they would like and take this value as input
- output "`Tickets booked`" and reduce the value stored in `tickets` if there are enough tickets available
- output "`Not enough tickets`" if there are not enough tickets available
- repeat the steps until there are no tickets left.

You must use either:

- OCR Exam Reference Language, or
- a high-level programming language that you have studied.

```
tickets = 500
```

```
x = int(input("how many tickets would you like to buy"))
output tickets = tickets - x
if tickets > 0
    print("Tickets booked")
else
    print("Not enough tickets")
tickets = 500
ticketcount = tickets - x
if tickets > 0 X:
    print("Tickets booked")
else
    print("Not enough tickets")
```

Not needed

deducts before the check so check will be after taking off;

Attempt of outputs

- No use of a loop
- No condition for loop

2

[6

END OF QUESTION PAPER