

Computer Science (J277)

J277/02, Computational thinking, algorithms and programming, June 2025

Please note that you may see slight differences between this paper and the original.

Candidates answer on the Question paper.

OCR supplied materials:
Additional resources may be supplied with this paper.

Other materials required:

- Pencil
- Ruler (cm/mm)

28
80

Duration: 90 mins

grade: 3

INSTRUCTIONS TO CANDIDATES

Write your name, centre number and candidate number in the boxes above. Please write clearly and in capital letters. Use black ink. HB pencil may be used for graphs and diagrams only.

Answer all the questions, unless your teacher tells you otherwise.

Read each question carefully. Make sure you know what you have to do before starting your answer.

Where space is provided below the question, please write your answer there.

You may use additional paper, or a specific Answer sheet if one is provided, but you must clearly show your candidate number, centre number and question number(s).

ADDITIONAL INFORMATION FOR CANDIDATES

The quality of written communication is assessed in questions marked with either a pencil or an asterisk. In History and Geography a *Quality of extended response* question is marked with an asterisk, while a pencil is used for questions in which *Spelling, punctuation and grammar and the use of specialist terminology* is assessed.

The number of marks is given in brackets [] at the end of each question or part question.

The total number of marks for this paper is 80.

The total number of marks may take into account some 'either/or' question choices.

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	✓		

[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 20 AND 23 then

print("accepted")

if num 10.5 then

print("not accepted")

endif

[3]

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

if -----

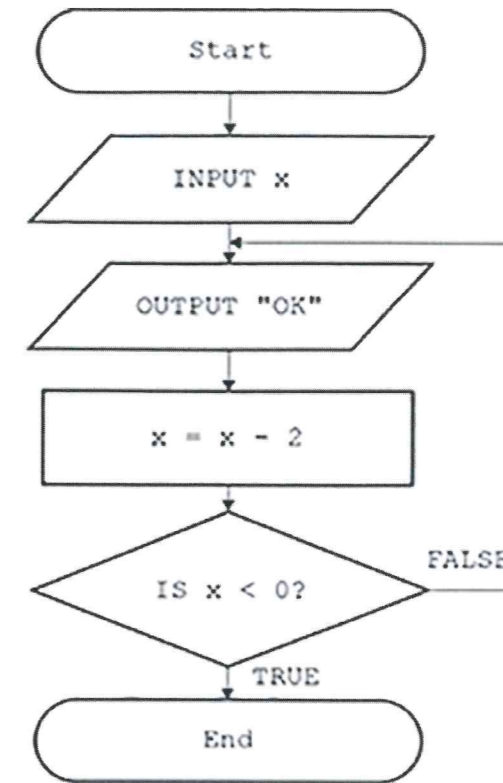
[1]

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	1
10	5

[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 LOOP is used when you know how many times you want to run the code.

2 WHILE LOOP is used when you do not know how many times going to run until it's false.

[4]

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

1 Selection Pseudocode

2 Flowchart

[2]

(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 error checker

Description It's going to highlight errors so the programmer knows what to change and it shows the code doesn't run

Tool 2 Colour code

Description It colour codes the code to show how ordered it is

[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 3 7 1 9 6 5 4

2 3 1 7 9 6 5 4

2 1 3 7 9 6 5 4

1 2 3 7 9 6 5 4

1 2 3 7 6 9 5 4

1 2 3 6 7 9 5 4

1 2 3 6 7 5 9 4

1 2 3 6 5 7 9 4

1 2 3 5 6 7 9 4

1 2 3 5 6 7 4 9

1 2 3 5 6 4 7 9

1 2 3 5 4 6 7 9

1 2 3 4 5 6 7 9

This is bubble sort.

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

2 1 3 1 9 6 5 4

1 1

2 3 7 1

0

[4]

(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	<input checked="" type="checkbox"/>
picks out the middle value to begin with	<input type="checkbox"/>
merges arrays of numbers together	<input type="checkbox"/>

A bubble sort compares pairs of values and then:

	Tick (✓) one box
always swaps them	<input type="checkbox"/>
swaps them if they are in the correct order	<input type="checkbox"/>
swaps them if they are in the incorrect order	<input checked="" type="checkbox"/>

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

	Tick (✓) one box
is declared with a fixed length	<input type="checkbox"/>
stores multiple values that cannot be changed while the program is running	<input checked="" type="checkbox"/>
can only be declared with a maximum of one-dimension (1D)	<input type="checkbox"/>

(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 is `false`.

i. Give the name of the searching algorithm described.

Merge sort

[1]

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

If everything is in order

[1]

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.

<code>data</code>	<code>for</code>	<code>if</code>
<code>myFile.close()</code>	<code>myFile.readLine()</code>	<code>input()</code>
<code>myFile.writeLine()</code>	<code>open</code>	<code>print</code>
<code>temp</code>	<code>txt</code>	<code>x</code>

`myFile = myFile myFile.readLine ("data.txt")`

`while NOT myFile.endOfFile()`

`temp = temp x`

`if int (input() input()) != 0`

`print print (temp)`

`endif`

`endwhile`

`myFile.close()`

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

[1]

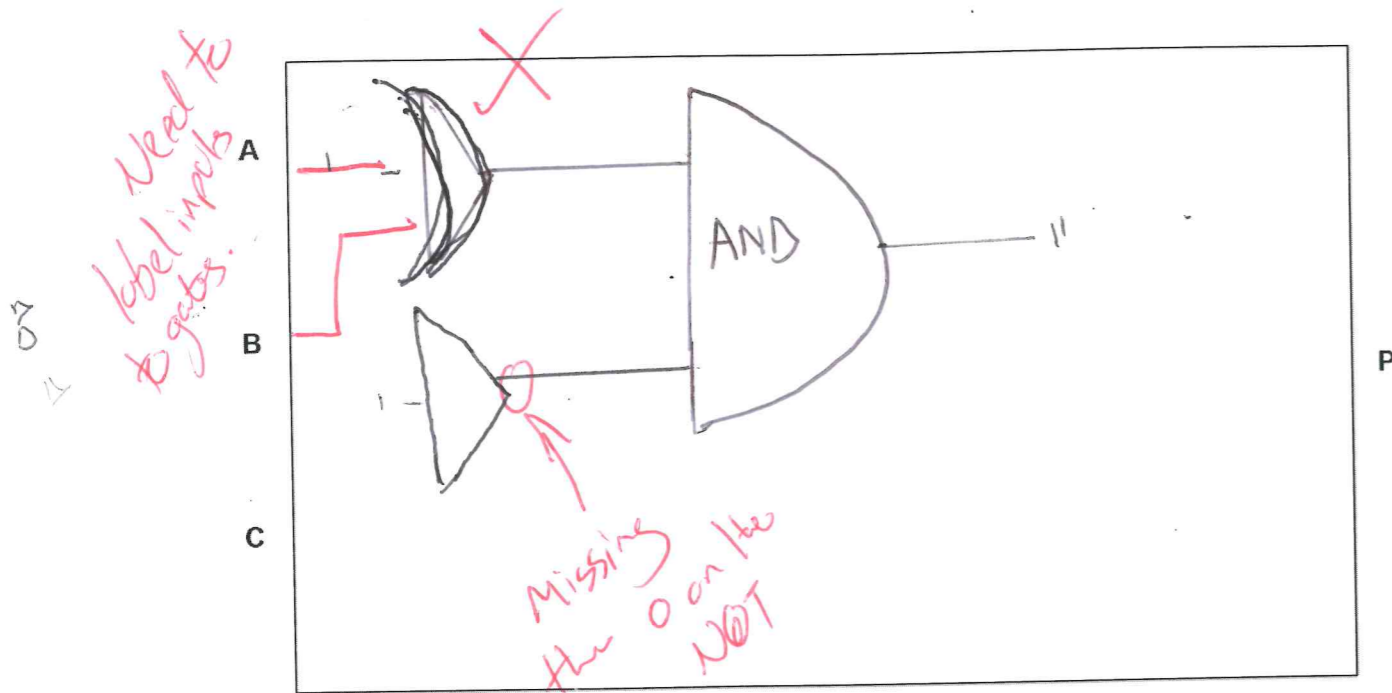
5(a) A security system protects online credit card payments. The system has three inputs:

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

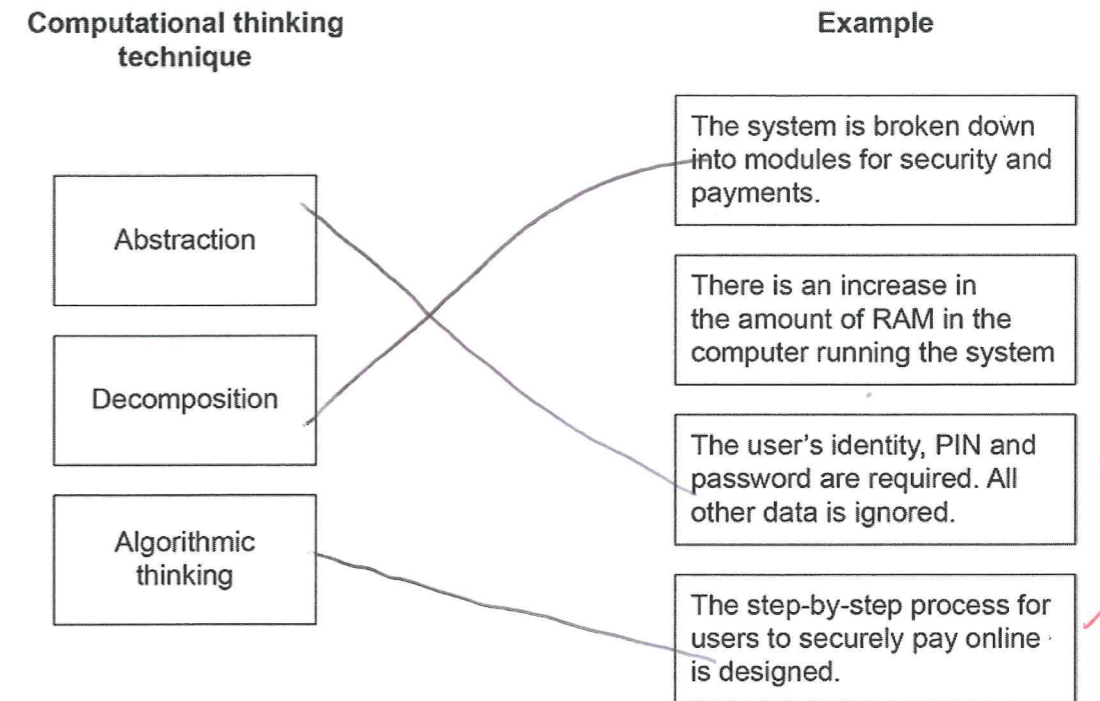


(b) Complete the truth table for $P = A \text{ AND } B$

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

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

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



(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.

num = str(input("is the PIN 4 characters in length"))
 if PIN is "1234" or "4321" then
 print("INVALID PIN")
 if PIN is not either "1234" or "4321"
 print("VALID PIN")
 end if

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
<code>passA and passB = input("enter password twice")</code>	
<code>input("enter your password") = passA</code> <code>input("enter your password again") = passB</code>	
<code>passA = input("enter your password")</code> <code>passB = input("enter your password again")</code>	✓
<code>"passA" = input("enter your password")</code> <code>"passB" = input("enter your password again")</code>	

[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.

`passA = input("enter your password")` ?

`pass B = input("enter your password again")`

`pass A = input("enter your password")`

`passA = cat`

`pass B = input("enter your password again")` 0

`pass B = cat`

`print("passwords match")` [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
WHERE Day = Saturday
```

[4]

ii. Identify one example of a record from TblMusic

Platinum

[1]

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

Field name	Data stored	Data type
Stage	The name of the stage	Real
SetLength	The length of each set in hours	format

[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.

```
def calculatePrice (num)
    num = input ("number of tickets to be purchased")
    print ("6")
    def: ("price for one x num of tickets")
    price = £60 x num
    total price: £360
    return price.
```

a good try, just structure issue. [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.

```
Total price = calculate price (3)
total ("total price x number of tickets")
£60 x 3
total price: £180
end if
```

[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
0	0	0	
1	1	0	
2	2	0	
3			33

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 The total act count is 60

[1]

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

END OF QUESTION PAPER

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`

repeated? →

```
NUM = INPUT("how many tickets would you like?") ✓
IF tickets wanted is 200 why 200?
    PRINT ("tickets booked") what if not 200 what is the check?
    tickets = ("how many tickets would you like?")
    IF 400 ticket is wanted
        PRINT ("NOT enough tickets")
    ticket = ("how many tickets would you like?")
    IF 300 tickets is wanted
        PRINT ("tickets booked")
    there are no tickets left.
end while.
```

where does the while start?

[6]