Computer Science GCSE (9–1) | OCR | J277

# Topic Tests

*for OCR GCSE Computer Science (J277)*

## *Component 2: Computational Thinking, Algorithms and Programming*

Update v1.1, 10 May 2022

zigzageducation.co.uk

POD 11239

# Contents

# Teacher's Introduction

Welcome to the OCR GCSE Computer Science Topic Tests, which have been written specifically for J277 specification, for first teaching September 2020 (assessment from 2022 onwards). This particular set of 10 tests covers the prescribed learning content for ***Component 2: Computational Thinking, Algorithms and Programming.***

These topic tests have been written with all new content for the J277 specification. These questions are designed to be used as end of unit tests, as unlike actual exam questions, they are designed to ask around the topic, supporting the knowledge required by the specification as opposed to sticking strictly to it. However, I took the decision to still write the questions in the style of real exam questions, which means they will better prepare students for the style of answers required. I also wrote the mark scheme to be like an exam mark scheme to help you as a teacher to understand where credit should and should not be awarded.

As such the following guidance should be followed when marking:

- Each bullet point is a single mark. You should not award 2 marks for points in one bullet point
- A "//" denotes an alternate answer for the same mark (not an extra mark) and a "/" an alternate phrasing
- Bold means this point must be made, e.g., for describing multiple cores **simultaneous** would be in bold as the student must describe this concept to get the mark
- Underline means the exact word must be in the answer (this is very rare)
- "…" at the end of a bullet point means this point must be made before the following points beginning with a "…" can be awarded. For example, when describing the best secondary storage to use in a camera and why, you would need to state a memory card before you can get awarded points for saying it is portable and can be moved from device to device.
- "…" at the beginning of a point without a preceding point with "…" at the end just means this point follows on logically. It can be awarded on its own, but it likely to come after the preceding point.
- Mark schemes that begin with "E.g." means that this question is very open and a mark scheme cannot cover every possible response and the given bullet points should be taken as indicative, awarding marks for any similar sensible response. For non- "E.g." questions, you should try to stick to the mark scheme.

I have also specified which Assessment Objective(s) (AOs) each question relates to, as this gives students a chance to understand what the AO's mean and recognise when they must apply their answers.

Each section has two question papers (*Set A* and *Set B*), which gives you options for showing progress, or using one when you teach and one when you do revision. Whilst the marks available for each sections varies – as some (e.g. programming) require more work – I have tried to keep the marks similar for the A and B tests and ensured that key concepts are covered by both.

**Remember!**

Always check the exam board website for new information, including changes to the specification and sample assessment material.

**Update v1.1, 10 May 2022**

- Corrected flow charts on pages 4, 9, 50, 53 and 54.

# 2.1 – Algorithms

## Test A (56 Marks)

QUESTION 1 – AO1

Define the term 'abstraction'.

.....................................................................................................................

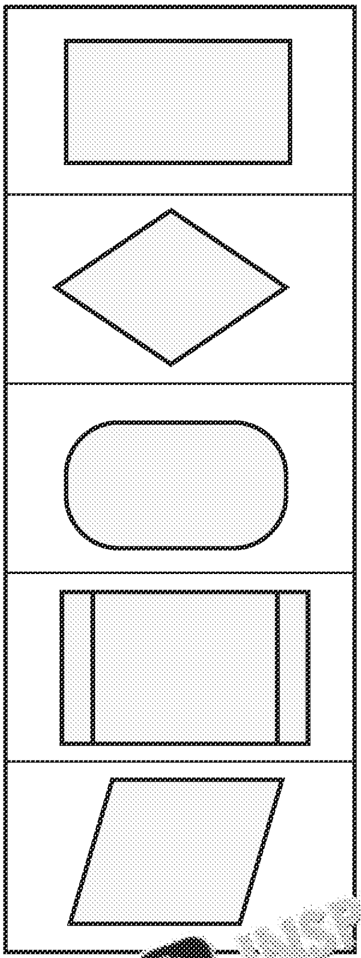
.....................................................................................................................

.....................................................................................................................

.....................................................................................................................

QUESTION 2 – AO1

Match the descriptions to the flow chart symbols.



Terminator

Process

Decision

Input/Output

Predefined process (subroutine)

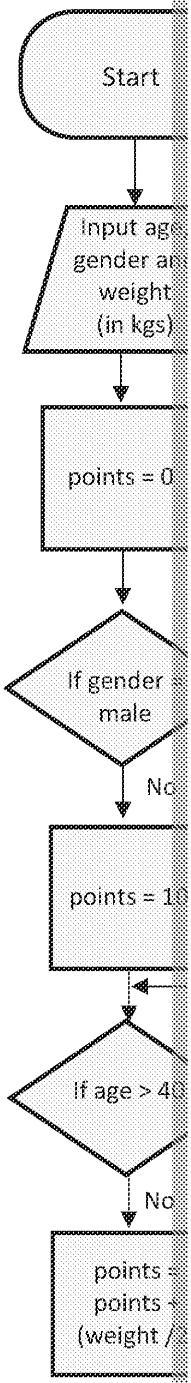
## QUESTION 3 – AO2

Figure 1 shows a flow chart used by a weight loss organisation to calculate memb gender and weight.

Calculate the points output for the following inputs.

| Age | Gender | Weight (kgs) | Points |
|---|---|---|---|
| 28 | Male | 110 | |
| 44 | Female | 75 | |
| 50 | Male | 90 | |
| 36 | Female | 6[illegible] | |

Figure 1


Start
Input age gender an weight (in kgs)
points = 0
If gender = male
No
points = 10
If age > 40
No
points = points + (weight /

### Question 4 – AO2

Gerard is writing a quiz program. As part of this, each question is asked a maximu

Design a flow chart that asks a single question, 'What is the capital of Ireland?', a the correct answer, 'Dublin'.

### Question 5 – AO2

Show the steps of a merge sort on the following list of numbers.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5 | 3 | 8 | 2 | 9 | 1 | 4 | 7 |

### Question 6 – AO3

Complete the following bubble sort algorithm.

```
function BubbleSort(sortList)
      sorted = ____________________
      while sorted == false
            sorted = true
            for sortCount = 0 to len(______________________
                  if sortList[sortCount] > sortList[so
                        ________________________ = sortList
                        sortList[sortCount] = sortList
                        sortList[sortCount + 1] = temp
                        ________________________ = false
                  endif
            next sortCount
      endwhile
      return sortList
endfunction
```

### Question 7 – AO1

In the following lines of code, explain the need for the `temp` variable.

```
temp = sortList [listIndex]
sortList[listIndex] = sortList[listIndex + 1]
sortList[listIndex + 1] = temp
```

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

........................................................................................................................................

### Question 8 – AO1

Figure 2 shows an algorithm.

Complete the following trace table and identify the purpose of the algorithm. You may not need to use all of the lines.

a = 18, b = 5

| a | b | a > b |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

**Figure 2**

```
a = int(input())
b = int(input())
while (a > b)
      a = a - b
endwhile
print(a)
```

Purpose: ..........................................................................................................................

### Question 9 – AO3

Figure 3 is an algorithm to show the 12 times table for any number. There are tw
– find the line numbers they are on, identify whether they are syntax errors or lo

**Figure 3**

```
01    num = int(input("enter the number))
02    for count = 0 to 12
03          ans = count + num
04          print(count + " x " + num + " = " + an
05    next count
```

Error 1: Line number: .................. Error type: ..........................................

Fix: ..............................................................................................................

Error 2: Line number: .................. Error type: ..........................................

Fix: ..............................................................................................................

### Question 10 – AO2

Bob's T-shirts is an e-commerce site that sells designer T-shirt labels.

Using abstraction, explain, with examples, how customers would be represented

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

### Question 11 – AO1

Describe the steps of an insertion sort.

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

..............................................................................................................

### Question 12 – AO2

Kacper is designing a number guessing game. The game will get a random numbe the number and tells them whether they are right or whether they are too high o guess the number within five attempts, they lose.

Design a flow chart for this game.

## Preview of Questions Ends Here

## 2.2 – Programming Fundamentals – Test A

|  | Answer | Mar |
|---|---|---|
| 1 | • Data that can change<br>• Stored in RAM<br>• Accessed through the use of a name / an identifier / a label | AO1 |
| 2 | > Greater than<br><= Less than or equal to<br>>= Greater than or equal to<br>== Equal<br>< Less than<br>!= Not equal to | AO1 |
| 3 | `gravi`[illegible] | AO2 |
| 4 | 06 | AO2 |
| 5 | 11 | AO2 |
| 6 | • = is the assignment operator<br>• == is a comparison operator<br>• = sets the value on the left with the value on the right<br>• == will compare the left and right values and return true or false | AO1 |
| 7 | `values = [4, 6, 3, 8, 5, 1, 2]`<br>**`min = values [0]`**<br>`for count = 1 to values.length`<br>`        if (min > values[count])`<br>`                `**`min = values[count]`**<br>`        endif`<br>`next count`<br>`print(min)` | AO3 |
| 8 | `age = int(input("What is your age?")`<br>`if(age >= 11 and age <= 19) then`<br>`print("You are of secondary school age.")`<br>`else if (age < 11) then`<br>`        print("You are of primary school age.")`<br>`else if (age > 19) then`<br>`        print("You are not of secondary school age.")`<br>`endif`<br>• Use else if<br>• Logic allows correct printing of both 'You are of secondary school age.' and 'You are not of secondary school age.'<br>• Logic allows the correct printing of 'You are of secondary school age.' | AO3 |

| | Answer | Mar |
|---|---|---|
| 9 | • While loop will run zero or more times<br>• Repeat loop will run one or more times<br><br>• Condition is at the start of the while loop<br>• Condition is at the end of the repeat loop | AO1 |
| 10 | `max = int(input())`<br>`count = 1`<br>`while (count <= max)`<br>`        print (max ^ 2)`<br>`        count = count + 1`<br>`endwhile`<br><br>• Uses while loop<br>• Sets count to 1 before loop<br>• While loop condition is correct for setting of count<br>• Count incremented inside loop after print | AO3 |
| 11 | `noOfResponse = int(input("How many values?"))`<br>`min = -1`<br>`max = -1`<br>`for responseCount = 1 to noOfResponse`<br>`        value = int(input("Please enter value"))`<br>`        if(max < value or max == -1) then`<br>`                max = value`<br>`        end if`<br>`        if (min > value  or min == -1) then`<br>`                min = value`<br>`        end if`<br>`next responseCount`<br><br>`print("Minimum value is " + str(min))`<br>`print("Maximum value is " + str(max))`<br><br>• Asks the user how many responses and stores value in variable<br>• Sets min and max to suitable starting values / first user input<br>• Iterates correct amount<br>• Gets value from user and stores it. Must be inside iterative structure.<br>• Selection checks whether max is less than input value, and, if it is, sets value to max. Must be inside iterative structure. Must deal with first value correctly.<br>• Selection checks whether min is greater than input value, and, if it is, sets value to min. Must be inside iterative structure. Must deal with first value correctly.<br>• Prints both min and max after iterative structure | AO3 |

| | Answer | Mar |
|---|---|---|

**12**

```
function FindRange(valA, valB, valC)
       min = 0
       max = 0
       if (valA > valB)
              max = valA
              min = valB
       else
              max = valB
              min = valA
       end if
       if (valC > max)
              max = valC
       end if
       if (valC < min)
              min = valC
       end if
       return max - min
end function
```

- Use a function definition (including end function)
- Three parameters passed in
- Min/max initialised
- Min/max successfully set off two values
- Min/max set off all three
- Max – min is returned

AO3

**13**

```
number = random(1, 100)
lives = 5
guess = -1
while (lives > 0 and guess != number)
       guess = int(input("Guess a number"))
       if (guess == number) then
              print("Well done")
       else if (guess > number) then
              print("Too high, try a lower number")
       else
              print("Too low, try a higher number")
       end if
       lives = lives - 1
endwhile
```

- Get a random number between 1 and 100
- Set lives and guess to initial values
- Use a loop to allow multiple guesses
- Use a loop to continue until number guessed or five guesses
- Use selection to see whether they are right and display suitable message
- Use selection to see whether they are too high and display suitable message
- Use selection to see whether they are too low and display suitable message
- Change lives by one

AO3