Chhetri Academy

chhetriacademy.com

GCSE COMPUTER SCIENCE

2.1

OCR J277 | Specification point 2.1.1

Computational thinking

Untiered

Answers and examiner comments

2	2022-2025	Untiered
question groups	available papers	tier

Using this booklet

1. Answer numbers match the companion question booklet.
2. Follow the original question number on marking tables.
3. Examiner comments are included where available.

Name _____ **Date** _____

Some entries use full source marking material in an appendix. Any unavailable mark schemes are identified beside the relevant answer.

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Mark</th><th>Guidance</th></tr>
<tr><td>1</td><td>(a)</td><td></td><td>1 mark per correct row

<table>
<tr><th>OCR Reference Language code</th><th>Selection</th><th>Iteration</th></tr>
<tr><td><pre>for i = 1 to 10
    print(i)
next i</pre></td><td></td><td>✓</td></tr>
<tr><td><pre>whilescore != 0
    playgame()
endwhile</pre></td><td></td><td>✓</td></tr>
<tr><td><pre>if playerHit() then
    score = score - 1
endif</pre></td><td>✓</td><td></td></tr>
<tr><td><pre>switch bonus:
    case 0:
        score = 9
    case 1:
        score = 7
    case 2:
        score = 5
endswitch</pre></td><td>✓</td><td></td></tr>
</table></td><td>4
(AO2 1b)</td><td>No mark given if both boxes in a row ticked.

Accept any response (ticks, crosses, etc) that clearly indicates candidate’s choice.</td></tr>
<tr><td>1</td><td>(b)</td><td></td><td>• <code>score = score + 1 // score +=1 // score++</code></td><td>1
(AO3 2b)</td><td>Allow other logically correct answers that result in score increasing by one and being overwritten. Do not accept <code>score + 1 / score = +1</code>

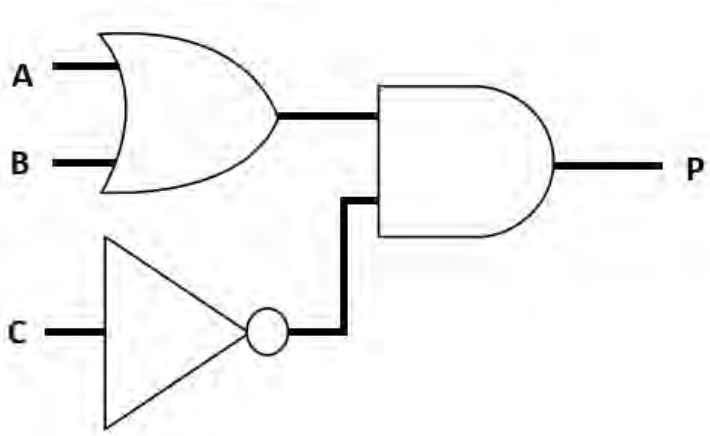
Accept valid structured English answers that refer to score increasing and overwriting the existing value by one. e.g. “score becomes/equals score plus one”

Ignore any superfluous code that does not affect the outcon</td></tr>
<tr><td></td><td>(c)</td><td></td><td>• Decomposition</td><td></td><td>Correct answer only. Ignore spelling errors.</td></tr>
</table>

			• Abstraction	**(AO1 1a)**	

<table>
<tr><th colspan="3">Question</th><th>Answer</th><th>Mark</th><th>Guidance</th></tr>
<tr><td>5</td><td>(a)</td><td></td><td>

Any use of an OR gate with 2 inputs
NOT C
Correctly joined by AND gate



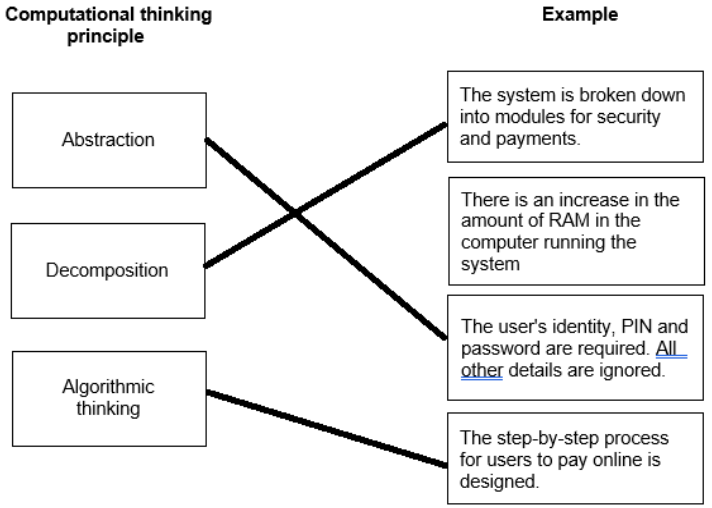
</td><td>3</td><td>Gate diagrams must be correct. NOT must include circle; AND / OR must not include circle.

FT for BP3 if appropriate (e.g. missing/wrong NOT gate, wrong gate used instead of OR)

Max 2 if extra gates or system is not logically correct.</td></tr>
<tr><td>5</td><td>(b)</td><td></td><td>

All input possibilities for A, B (00, 01, 10, 11), does not have to be in order…
…Correct output for P (only 1 when A and B are both 1, 0 otherwise).

</td><td>2</td><td>P = A AND B
<table>
<tr><th>A</th><th>B</th><th>P</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
Accept T/F, Yes/No, ticks/crosses, etc.</td></tr>
</table>

Question			Answer	Mark	Guidance
5	(c)		• Correct link for abstraction to third example • Correct link for decomposition to first example • Correct link for algorithmic thinking to bottom example	3	No mark for more than one connection from a box. 

Question			Answer	Mark	Guidance

5	(d)		• Input **and** storage/use of PIN
			• Attempt at selection / condition controlled loop…
			• …Correctly checks for 4 digits
			• …Correctly checks for **both** 1234, 4321
			• Correct output of “INVALID PIN” after sensible attempt at <u>all three</u> checks
			• Correct output of “VALID PIN” after sensible attempt at <u>all three</u> checks, **only** when PIN is fully valid and no contradictory output (e.g. printing “invalid PIN” as well)
			
			If diagram/flowchart given, ignore shapes if incorrect and mark text inside as algorithm.
				6	**<u>Example code</u>**
					`pin = input("Enter PIN")`
					`if pin.length == 4 and pin != "1234" and pin != "4321" then`
					`  print("VALID PIN")`
					`else:`
					`  print("INVALID PIN")`
					
					Note - | or || means OR in some languages. AND is &.
					
					BOD input and use/comparison of PIN as integer. No need for quotation marks around 1234 / 4321. Allow equivalent outputs.
					
					BP3 and 4 can be combined or as separate checks. **If combined, must logically work**. BP3 and 4 can be checks for positive or negative. Check must refer to variable for **both** comparisons.
					
					BP5 given if **invalid** output (once or multiple times) for all invalid PINs even if valid also output. BP6 only given if **valid** is output as the only output.
					
					If sensible attempt at BP3 and BP4 (but incorrect), FT to BP5 and 6.
					
					Section A so candidates can respond in pseudocode, structured English, flowcharts, etc. However, must not simply be a repeat of the question.

Question 1 (a)

1 (a) Tick (✓) **one** box in each row to identify whether the OCR Reference Language code given is an example of selection or iteration.

OCR Reference Language code	Selection	Iteration

`for i = 1 to 10`
`  print(i)`
`next i`		

`while score != 0`
`  playgame()`
`endwhile`		

`if playerHit() then`
`  score = 0`
`endif`		

`switch bonus:`
`  case 0:`
`  score = 9`
`  case 1:`
`  score = 7`
`  case 2:`
`  score = 5`
`endswitch`		

[4]

Candidates appeared to struggle with this question. In particular the use of switch/case was not well understood. This may be because some high-level languages such as Python have not traditionally supported this. The recent update to Python 3.10 introduces this construct. Centres who use Python may want to consider upgrading their installation to allow practical implementation of the switch/case construct.

Question 1 (b)

(b) Write pseudocode to increment the value held in the variable `score` by one.

.....

..... **[1]**

Most candidates were able to temporarily add one to the value of score (by using code such as `score + 1`). Fewer were able to assign this new value to the variable of score and therefore fully answer the question. Answers such as `score = score + 1`, `score++` or `score += 1` were all accepted, as were any longer responses that used intermediate variables. As long as the value of `score` ended up being incremented by one, examiners were instructed to give credit.

Assessment for learning

One notable response that was **not** allowed was `score =+1`. This statement assigns the value of 'positive 1' to score, overwriting the previous value held.

This is a good example of the precision required to gain marks. Candidates with practical programming experience are more likely to recognise this.

Question 1 (c)

(c) State the name of each of the following computational thinking techniques.

Breaking a complex problem down into smaller problems.

.....

.....

Hiding or removing irrelevant details from a problem to reduce the complexity.

.....

.....

[2]

This question was answered well by the majority of candidates.

Question 5 (a)

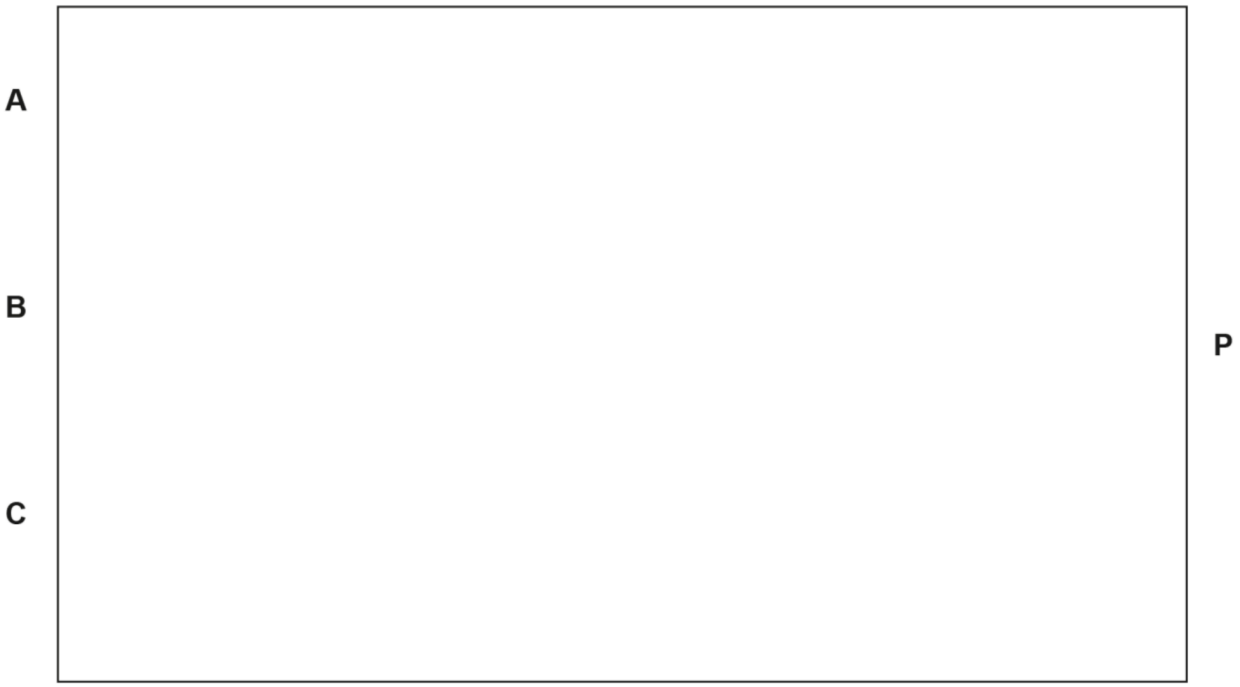
5 A security system protects online credit card payments. The system has three inputs:

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.

(a) Draw the logic circuit for the security system



[3]

This question was extremely well answered by candidates, showing excellent understanding of constructing logic gate diagrams from a real-world scenario.

Misconceptions or mistakes were rare, but where they did occur they tended to be around shapes of gates. Candidates should be aware that the circle on the NOT gate is crucial (as otherwise a triangle with no circle is simply a buffer) and that AND and OR gates do not need circles, or they would instead of NAND or NOR gates.

Examiners are instructed to mark the shapes of the gates and ignore any annotation, so it is crucial that candidates can accurately draw shapes that are clearly AND and OR gates.

Question 5 (b)

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

A	B	P

[2]

This question was also extremely well answered. Most candidates filled in columns A and B with every possible combination of inputs. 1s and 0s were the expected options, but Yes/No and True/False were also accepted if given by candidates.

Inputs do not have to be in any particular order, but best practice to start with 00 and then essentially count up in binary, through 01, 10 then 11. This means that where larger systems are presented (perhaps with 3 or 4 inputs), no input combination is missed.

The final column (P) should have contained the output for the system, and with P = A AND B, this means that only the row with A and B as both 1 should have had a 1 output, with every other output being 0.

Question 5 (d)

(d) As part of the 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.

.....

.....

.....

.....

.....

..... **[6]**

As a question in Section A, candidates are free to use a high-level programming language, flowcharts or structured English. Pleasingly, the vast majority attempted to use a high-level programming language and this proved to be very successful for many.

Candidates proved able to take input as a string, to use in-built length functions to check for length and to output messages based on these checks.

The major misconception that affected many students related to checking two invalid PINs, 1234 or 4321. Where two checks are done within a single IF statement it is important that each condition **must** be a valid comparison and therefore include the variable to check against for both comparisons.

This is the same misconception highlighted on Question 1 (b) (i).

Misconception

Selection statements using two conditions joined together with a Boolean condition such as OR must make sure that both conditions are logically valid comparisons. In many cases, this will mean referencing a variable twice, once in each condition.

For example, `if PIN == "1234" OR "4321"` is incorrect because the second condition is not a valid comparison – what is being compared to "4321" in order for the condition to be evaluated as True? The correct statement would be `if PIN == "1234" OR PIN == "4321"`.

Section B overview

Section B consists of multiple sub-questions all centred around one scenario (in this case, a music festival). Where algorithms are asked for, candidates must respond using either OCR Exam Reference Language or a high-level programming language that they are familiar with. Where responses are presented using flowcharts or structured English in Section B, no marks are given.

However, it is important for centres and candidates to understand that examiners are not told to penalise responses because they would not work in a particular language. Candidates do not have to state which language they are writing in, and so answers are marked for their logical correctness and consistency. This includes where responses combine features from multiple high-level languages.

Answers are never marked down for use of case.