Qualification Accredited

OCR
Oxford Cambridge and RSA

GCSE (9-1)

Examiners' report

COMPUTER SCIENCE

J277

For first teaching in 2020

J277/02 Summer 2022 series

Version 1

ocr.org.uk/computerscience

Contents

Introduction

Our examiners' reports are produced to offer constructive feedback on candidates' performance in the examinations. They provide useful guidance for future candidates.

The reports will include a general commentary on candidates' performance, identify technical aspects examined in the questions and highlight good performance and where performance could be improved. A selection of candidate answers are also provided. The reports will also explain aspects which caused difficulty and why the difficulties arose, whether through a lack of knowledge, poor examination technique, or any other identifiable and explainable reason.

Where overall performance on a question/question part was considered good, with no particular areas to highlight, these questions have not been included in the report.

A full copy of the question paper and the mark scheme can be downloaded from OCR.

Advance Information for Summer 2022 assessments

To support student revision, advance information was published about the focus of exams for Summer 2022 assessments. Advance information was available for most GCSE, AS and A Level subjects, Core Maths, FSMQ, and Cambridge Nationals Information Technologies. You can find more information on our website.

Paper 2 series overview

This is the first examination series for the new J277 specification.

J277/02 (Computational thinking, algorithms and programming) is one of two examination components for GCSE Computer Science. This component focuses on:

- algorithms
- programming fundamentals
- producing robust programs
- boolean logic
- programming languages and Integrated Development Environments

To do well on this paper candidates need to be comfortable with writing, completing and using algorithms using pseudocode and/or flowcharts. This may involve applying their knowledge to unfamiliar contexts. In Section B, candidates are asked to provide answers using either OCR Exam Reference Language or a high-level language that they have studied.

Where candidates had extensive practice of producing and completing algorithms using a high-level language in classroom situations, this clearly allowed them to answer questions on this paper more confidently.

It is a requirement that all centres allocate sufficient time for practical programming tasks to be completed by candidates.

Centres are encouraged to be aware of the contents of the specification. Section 3c shows details of OCR Exam Reference Language. These conventions are used when setting examination questions. Recognising these conventions will help candidates to successfully understand and access examination questions.

Candidates who did well on this paper generally did the following:	Candidates who did less well on this paper generally did the following:
• understood how sorting and searching algorithms are applied to given data sets and were able to be precise in describing the steps taken • were able to write algorithms and program code to solve problems, using iteration and selection successfully to meet the stated requirements • could identify errors in code and were able to rewrite or refine these to provide working solutions • were clear on the implementation and use of functions that accept parameters and return a value.	• used non-standard flowchart symbols • missed off key inputs and decisions when designing algorithms • described outcomes of sorting and searching algorithms rather than the precise steps taken • were unclear about the logical difference between AND and OR when validating inputs • used unsuitable mathematical symbols (such as x for multiplication or ≥ for comparison) in Section B. These symbols are not used in either OCR Exam Reference Language or any popular high-level language.

Section A overview

Section A consists of multiple questions and scenarios. Candidates are free to write algorithms in any suitable way. This may include using flowcharts, structured English, pseudocode or a high-level language. The majority of candidates who scored highly tended to use a high-level language consistently.

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 2 (a) (i)

2 A fast food restaurant offers half-price meals if the customer is a student or has a discount card. The offer is not valid on Saturdays.

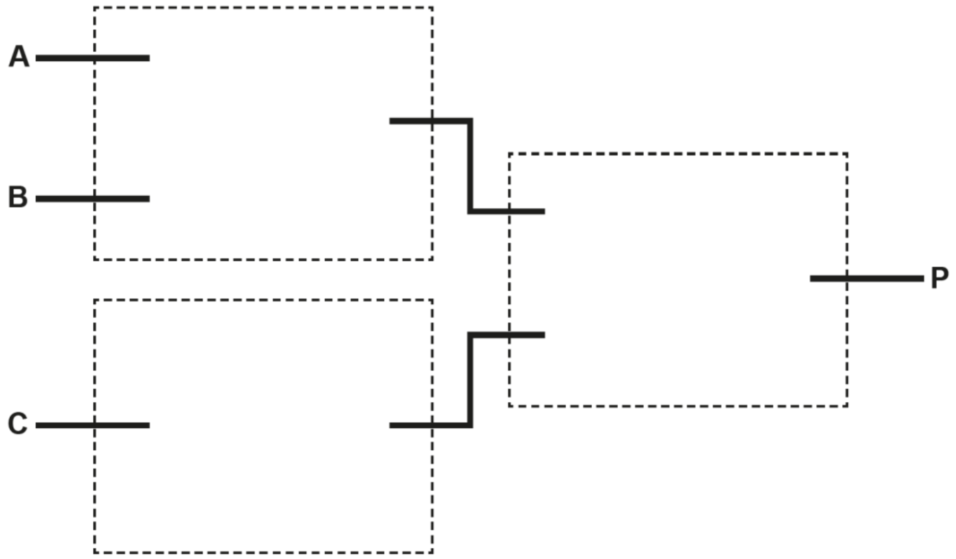
A computer system is used to identify whether the customer can have a half-price meal.

The table identifies the three inputs to the computer system:

Input	Value
A	Is a student
B	Has a discount card
C	The current day is Saturday

(a) The logic system **P = (A OR B) AND NOT C** is used.

(i) Complete the following logic diagram for **P = (A OR B) AND NOT C** by drawing one logic gate in each box.



[3]

This question was answered extremely well. Candidates were familiar with the logic symbols required for each gate.

A few candidates missed the circle from the NOT gate symbol (which turned this into a buffer instead) or added a circle to the AND and OR gates. This turned them into NAND and NOR gates. These logic gates are not on the GCSE specification, but are covered at A Level. Teachers may find it beneficial to briefly discuss these in order to anticipate mistakes like these.

A small number of candidates drew indistinct logic gate symbols and named them instead, such as a rectangle with AND written in; this was not credited with a mark.

Question 2 (a) (ii)

(ii) A truth table can be produced for this logic circuit.

Describe the purpose of a truth table.

....

....

....

.... **[2]**

Question 2 (a) (iii)

(iii) State how many rows (excluding any headings) would be required in a truth table for the logic expression:

P = (A OR B) AND NOT C

.... **[1]**

It is clear that candidates have experienced and used truth tables with logic circuits. However, the majority were not able to describe the truth table's purpose with precision.

A truth table defines the expected outputs for a logic circuit depending on the inputs given. Furthermore, the truth table covers all possible permutations of inputs.

A logic circuit with three inputs (A,B and C) will have **8** possible rows in a truth table to cover the 8 possible ways that True and False values for three inputs can be arranged.

Misconception

The number of rows in a truth table depends on the number of inputs. The number of rows can be given by 2^x, where x is the number of inputs. Therefore, a truth table with 3 inputs would have 2^3 rows, or 2 x 2 x 2 = 8 rows.

Exemplar 1

To show what output happens for every variation of inputs, and check they are all what was intended

Here the candidate has achieved both marks available; it is clear that every variation of input is included in the truth table and the output is very clearly linked to these inputs.

Question 2 (b)

(b) The restaurant needs an algorithm designing to help employees work out if a customer can have a half price meal or not. It should:

- input required data
- decide if the customer is entitled to a discount
- output the result of the calculation.

Design the algorithm using a flowchart.

[5]

Most candidates who answered this question were comfortable using the correct flowchart symbols listed in the specification. A mark was available for including suitable start / end symbols and connecting all other symbols, so even candidates who may struggle should feel confident of being able to access some marks.

Marks were dropped when responses did not include suitable inputs to the algorithm. The first bullet point in the question stem was clear that these were required.

Candidates achieving lower scores tended to group decisions so that any processing was removed (such as "can they have a discount?"). More successful responses decomposed the problem and used a succession of smaller decisions ("do they have a discount card?", "are they a student?", "is it Saturday?"). These decisions then point towards the correct outputs.

Question 2 (c)

(c) The restaurant adds a service charge to the cost of a meal depending on the number of people at a table. If there are more than five people 5% is added to the total cost of each meal.

Customers can also choose to leave a tip, this is optional and the customer can choose between a percentage of the cost, or a set amount.

Identify **all** the additional inputs that will be required for this change to the algorithm.

....

....

....

.... **[2]**

This question type is new to the J277 specification and asked candidates to list inputs that will be required as part of the planning stage for an algorithm. Successful candidates were able to identify the raw data needed from the user in order to be able to solve the problem.

Unsuccessful responses tended to simply rewrite the question or miss out key information.

Question 2 (d) (i)

(d) Each member of staff that works in the restaurant is given a Staff ID. This is calculated using the following algorithm.

```
01  surname = input("Enter surname")
02  year = input("Enter starting year")
03  staffID = surname + str(year)
04  while staffID.length < 10
05      staffID = staffID + "x"
06  endwhile
07  print("ID " + staffID)
```

(i) Define the term **casting and** give the line number where casting has been used in the algorithm.

Definition ....

....

Line number ....

[2]

Many candidates correctly defined casting as changing data from one data type to another. Some candidates defined this term as changing a variable from an integer to a string, which is only one example of what can be done and not a definition.

The majority of candidates then gave the correct line number (line 03) for there this was shown the example code given.

Question 2 (d) (ii)

(ii) Complete the following trace table for the given algorithm when the surname "Kofi" and the year 2021 are entered.

You may not need to use all rows in the table.

Line number	`surname`	`year`	`staffID`	Output
01	Kofi			
02		2021		

[4]

This question asked candidates to trace through a given algorithm to show the value of three variables at various points in the algorithm.

The algorithm itself was relatively simple. It used condition-controlled iteration to repeat while the length of the username was less than 10 characters.

Most candidates gained the first 2 marks for the initial changes to `staffID`. However few candidates were able to trace through the iteration and conclude that the final username should end up as **`ID Kofi2021xx`**.

Marking this question considered the spaces within the username at various points. The algorithm results in one space only, in between **`ID`** and **`Kofi2021xx`**. Where extra spaces appeared or were missed, this was penalised. However, examiners were instructed to give clear benefit of doubt, and to only do this if the space was clearly present/missing.

It is important to understand that "`ab`" and "`a b`" are two strings that are not the same. This level of precision should be encouraged within GCSE Computer Science. Experience of practical programming will help reinforce the impact of spaces within programming and algorithms.

Question 3 (a)

3 A program stores the following list of positive and negative numbers. The numbers need sorting into ascending order using a merge sort.

45	12	–99	100	–13	0	17	–27

(a) The first step is to divide the list into individual lists of one number each. This has been done for you.

Complete the merge sort of the data by showing each step of the process.

45	12	–99	100	–13	0	17	–27

[3]

Most candidates were able to correctly demonstrate the required steps of the merge sort algorithm. This involves merging the set of 8 individual lists into 4 lists of size 2, then 2 lists of size 4, then 1 list of size 8. At each point, each list produced must be in order. It is this process of merging lists together that sorts the values.

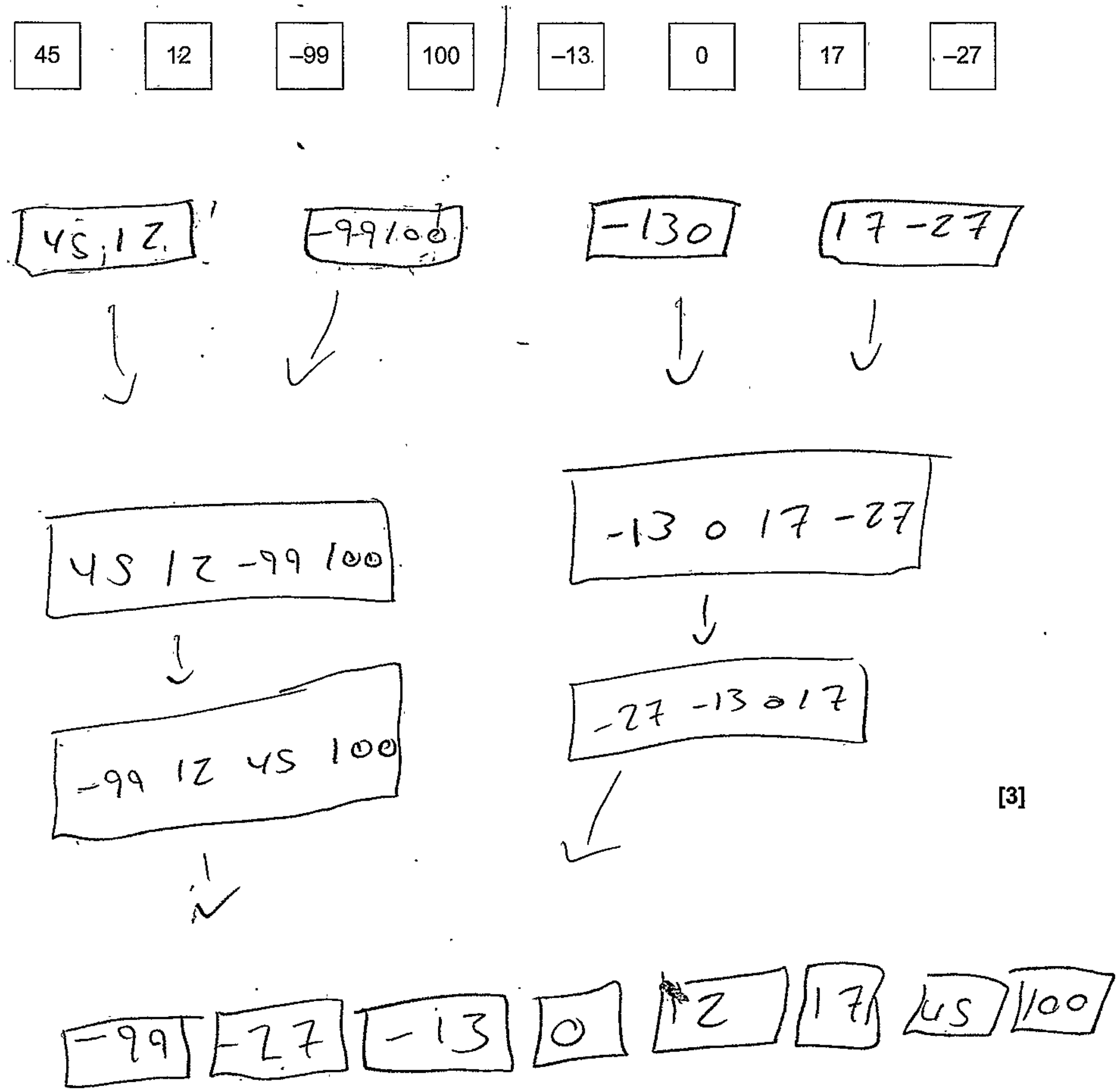
No mark was given simply for showing the final list sorted unless the previous step(s) had been followed; any candidates an alternative sorting algorithm (for example a bubble sort) would achieve 0 marks even though the final outcome is the same.

Misconception

Many candidates showed merged lists which were out of order to start with, and then sorted these merged lists. This is highly inefficient and not what the standard merge sort algorithm describes.

In merge sort, the process of merging lists together produces the sorted list. There does not need be any second step to sort values as they should already be in order.

Exemplar 2



[3]

This exemplar shows the misconception detailed previously. Here, the candidate firstly merges together the individual lists into 4 lists of size 2, but these lists are not all in order (e.g. 17, -27 is incorrect).

The second step then merges these lists together into 2 lists of size 4, but again, these are not in order. An unnecessary next step is then shown where the lists are then sorted in place.

Question 3 (b)

(b) Once the numbers are in order, a binary search can be run on the data.

Describe the steps a binary search will follow to look for a number in a sorted list.

....

....

....

....

....

.... **[4]**

This question asks candidates to describe how a binary search would look for a number in a sorted list. Most candidates answered this well. The most common mark for this question was 3 out of 4.

The majority of candidates clearly understood that the middle value is chosen and compared against the value being searched for. However, almost all candidates then went on to discuss which half of the list should be discarded or focused on based on this comparison.

One possibility is that the middle value is the number we are searching for. Few candidates discussed this possibility. As the process repeats, this check is important if the number is ever to be found!

Assessment for learning

When discussing a binary search there are **three** possibilities when comparing the middle value to the number being searched for.:

1. the middle value is higher than needed (in which case the left half of the list should be focused on)
2. the middle value is lower than needed (in which case the right half of the list should be focused on)
3. the middle value is equal to the value needed (in which case the value has been found and the process can stop.

Most responses only considered the first two of these possibilities. This is worth discussing when teaching this topic.

Question 3 (c)

(c) A linear search could be used instead of a binary search.

Describe the steps a linear search would follow when searching for a number that is **not** in the given list.

....

....

....

.... **[2]**

Responses to this question generally lacked precision.

The question specifically asks about searching for a number that is **not** in the list. However, many candidates discussed stopping when the value had been found. This is clearly not possible given the scenario.

Many responses discussed checking each number but did not state whether this would be in a particular order. Some candidates discussed checking values randomly, or even from the right down to the left.

For this questions, candidates may have known that a linear search starts at the left most number and then checks each value in the list in order until reaching the end of the list. However, few responses stated this or anything close to this.

This level of precision (and conversely, not assuming that steps are obvious) should be encouraged within GCSE Computer Science.

Question 4 (a)

4 Jack is writing a program to add up some numbers. His first attempt at the program is shown.

```
a = input("Enter a number")
b = input("Enter a number")
c = input("Enter a number")
d = input("Enter a number")
e = input("Enter a number")
f = (a + b + c + d + e)
print(f)
```

(a) Give **two** ways that the maintainability of this program could be improved.

1 ....

....

2 ....

.... **[2]**

This question asked about maintainability. It was important that a candidate's response referred to the code given.

"Give two ways that maintainability of **a program** could be improved" is a different question than the one asked.

Because of this, responses that were accepted must genuinely improve the maintainability of the program shown. One very common wrong response was indentation; although this would be useful generically, there was no code presented that would benefit from being indented.

Many candidates were able to give responses such as use of comments and sensibly named variables that would genuinely improve the given program. Where candidates described putting the given code inside a newly defined subroutine. This response was credited. However some responses simply said "use a subroutine". This was not enough.

Question 4 (b) (i)

(b) Jack's program uses the addition (+) arithmetic operator. This adds together two numbers.

(i) State the purpose of each of the arithmetic operators in the table.

Arithmetic operator	Purpose
*	
/	

[2]

This question was answered well by the majority of candidates.

Question 4 (b) (ii)

(ii) Complete the description of programming languages and translators by writing the correct term from the box in each space.

continues	crashes	debugging	error	executable
high-level	interpreter	language	low-level	many
no	one	stops	with	without

Jack writes his program in a .... language. This needs to be translated into assembly or machine code before it can be executed. This is done using a translator.

One type of translator is an interpreter. This converts one line of code and then executes it, before moving to the next line. It .... when an error is found, and when corrected continues running from the same position. This translator is helpful when debugging code.

A second type of translator is a compiler. This converts all of the code and produces an error report. The code will not run until there are .... errors. The .... file produced can be run .... the compiler.

[5]

This question was answered well by the majority of candidates.

Question 4 (c)

(c) Jack decides to improve his program. He wants to be able to input how many numbers to add together each time the algorithm runs, and also wants it to calculate and display the average of these numbers.

Write an algorithm to:

- ask the user to input the quantity of numbers they want to enter and read this value as input
- repeatedly take a number as input, until the quantity of numbers the user input has been entered
- calculate and output the total of these numbers
- calculate and output the average of these numbers.

....

....

....

....

....

.... **[6]**

As this question appeared in Section A, candidates are free to respond in any suitable way, including using flowcharts, structured English, pseudocode or a high-level language. The majority of candidates who scored highly tended to use a high-level language consistently.

The question is already decomposed for candidates and many were able to use these bullet points to build a solution that achieved the majority (or all) marks available.

Where mistakes were made, these tended to be with repeating code. For example, many candidates repeated the process of adding values up without repeatedly asking for a new number (and therefore continually adding the same number).

Other candidates used condition-controlled iteration to repeat the process but did not make sure that the condition being evaluated ever returned a `False` value, therefore repeating infinitely.

Where a mistake was made in one section (such as with iteration), examiners were instructed to use FT (follow through) marks where possible if later sections were logically constructed. This made sure that marks could be given where appropriate.

A few candidates were not able to access many of the marks available in this question, suggesting that they would benefit from more practical programming time in lessons.

Section B overview

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

Candidates do not have to state which language they are using and so responses are marked for their logical correctness and consistency.

It is important for centres and candidates to understand that examiners are told to **not** penalise responses because they would not work in a particular language.

Question 5 (a) (i)

5 Customers at a hotel can stay between 1 and 5 (inclusive) nights and can choose between a basic room or a premium room.

(a) A typical booking record is shown in the table:

`firstName`	Amaya
`surname`	Taylor-Ling
`nights`	3
`room`	Premium
`stayComplete`	False

(i) State the most appropriate data type for the following fields:

Nights ....

Room ....
[2]

Question 5 (a) (ii)

(ii) Give the name of **one** field that could be stored as a Boolean data type.

.... **[1]**

These questions tested candidates' knowledge of data types and it was clear that this knowledge was well understood. The majority of candidates were able to correctly identify suitable data types in section (i) and identify `stayComplete` as the field that would be stored as a Boolean data type.

Question 5 (a) (iii)

(iii) Booking records are stored in a database table called `TblBookings`.

The following SQL statement is written to display all customer bookings that stay more than one night.

```
SELECT ALL
FROM TblBookings
IF Nights < 1
```

The SQL statement is incorrect.

Rewrite the SQL statement so that it is correct.

....

....

....

....

....

.... **[4]**

This question tested candidates' ability to refine and rewrite incorrect code given to them. It is important to note that although the SQL statement as a whole is incorrect, not all components are incorrect; in this case, the `FROM` clause is correct and candidates who made no change to this line were credited.

`SELECT ALL` is invalid SQL and should have been written to instead include all fields from the table, separated by commas. `SELECT *` was equally accepted as a suitable response.

`IF` is not a valid SQL keyword and needs to be replaced with `WHERE`. The criteria for this statement was also incorrect. The comparison symbol is incorrect and should read `Nights > 1`.

Most candidates gained some marks on this question. The most common response corrected the criteria and not modifying the `FROM` clause.

Question 5 (b) (i)

(b) When a new booking is recorded, the details are entered into a program to validate the values. The following criteria are checked:

- `firstName` and `surname` are not empty
- `room` is either "basic" or "premium"
- `nights` is between 1 and 5 (inclusive).

If any invalid data is found "NOT ALLOWED" is displayed.
If all data is valid "ALLOWED" is displayed.

(i) Complete the following program to validate the inputs.

You must use **either**:
- OCR Exam Reference Language, **or**
- a high-level programming language that you have studied.

```
firstName = input("Enter a first name")
surname = input("Enter a surname")
room = input("Enter basic or premium")
nights = input("Enter between 1 and 5 nights")
stayComplete = False
```

....

....

....

....

....

.... **[5]**

This question stretched the understanding of even highly-achieving candidates and it was not uncommon to see low scoring responses.

Misunderstanding of Boolean operators (`AND` and `OR`) within selection (`IF`) statements was something that affected a lot of candidate responses.

As this question was in Section B, candidates needed to respond in OCR Exam Reference Language or a high-level language. Responses must be logically correct to gain the marks. As each check is two individual checks that both need to pass, responses can quickly get relatively complicated.

As can be seen from the mark scheme, advice and examples were given to examiners to make sure that candidates who were able to successfully navigate this logic chain were credited.

Misconception

Checking whether a room is either basic or premium can be done in multiple ways. Candidates can either check for the positive (i.e. check that it is either basic or premium) or check that for the negative (i.e. check whether it is something else). However, there are many common errors that were seen :

- `IF room == "basic" or "premium"` is **incorrect** as the second part of the statement is not evaluated against anything. This was perhaps the most common mistake.
- `IF room == "basic" or room == "premium"` is **correct** and checks for validity.
- `IF room == basic or room == premium` is **incorrect** as the lack of string delimiters means that basic and room would be treated as variables rather than strings.
- `IF room != "basic" or room != "premium"` is also **incorrect**. This checks for invalid input but because **or** is used, only one condition needs to be True for the whole statement to be True. This means that if **basic** is entered, it would be classed as invalid (as it isn't premium) and vice-versa. There is no way for any entry in this example to be classes as valid.
- `IF room != "basic" and room != "premium"` is **correct**. This checks for invalid inputs but needs both conditions to be True.

The same explanation follows for the other two necessary checks.

Exemplar 3

```
if first Name == "" OR surname == "" then
        print("NOT ALLOWED")
else
    if room != "basic" AND room != "premium"
    then
            print("NOT ALLOWED")
    else
            if nights<1 OR nights>5 then
                print("NOT ALLOWED")
            else
                print("ALLOWED")
            endif
    endif
endif
```

This exemplar shows a fully correct response. The candidate checks for invalid responses and correctly uses Boolean operators to check multiple criteria at each step. If any check returns True, "Not allowed" is printed and the program ends. Efficient use of `if … else …` means that the next check only proceeds if the previous check returns False.

If all three checks return False, the final else is triggered to print "Allowed".

It must be noted that this is only one way of achieving full marks. An equivalent program that checks for valid responses at each turn would also be possible. Candidates should be encouraged to use whatever structure they feel is sensible. If a response can logically be followed then it will achieve high marks.

Question 5 (b) (ii)

(ii) Complete the following test plan to check whether the number of nights is validated correctly.

Test data (number of nights)	Type of test	Expected output
2		ALLOWED
	Boundary	ALLOWED
	Erroneous / Invalid	NOT ALLOWED

[3]

This question was answered well by the majority of candidates.

Question 5 (c) (i)

(c) A Basic room costs £60 each night. A Premium room costs £80 each night.

(i) Create a function, `newPrice()`, that takes the number of nights and the type of room as parameters, calculates and returns the price to pay.

You do **not** have to validate these parameters.

You must use **either**:
- OCR Exam Reference Language, **or**
- a high-level programming language that you have studied.

....

....

....

....

....

.... **[4]**

Defining functions appeared to be a concept that candidates did not fully understand.

Where a candidate did not attempt to define a function and instead simply calculated the price needed, very few (if any) marks were available.

Successful responses could have been constructed from any suitable function definition keyword such as `function` (OCR ERL, VB, JavaScript, etc), `def` (Python) or others. Answers in C#, Java or other languages referring to methods were also accepted.

Question 5 (c) (ii)

(ii) Write program code, that uses `newPrice()`, to output the price of staying in a Premium room for 5 nights.

You must use **either**:
- OCR Exam Reference Language, **or**
- a high-level programming language that you have studied.

....

....

....

.... **[3]**

Even if candidates were not able to create a function, this question was independent to (i) and so marks were available for simply using the function to output a value.

Candidate found this question challenging. Many candidates called the function but most did not understand that the room type was a string and so required string delimiters (e.g. quotation marks) around the parameter.

Where candidates defined local variables, assigned the values needed to the variables and then passed these into the function call were accepted.

Question 5 (d)

(d) The hotel has nine rooms that are numbered from room 0 to room 8.

The number of people currently staying in each room is stored in an array with the identifier `room`.

The index of `room` represents the room number.

Array `room`

Index	0	1	2	3	4	5	6	7	8
Data	2	1	3	2	1	0	0	4	1

The following program counts how many people are currently staying in the hotel.

```
for count = 1 to 8
    total = 0
    total = total + room[count]
next count
print(total)
```

When tested, the program is found to contain **two** logic errors.

Describe how the program can be refined to remove these logic errors.

....

....

....

....

....

.... **[2]**

This question acted as an excellent discriminator. The majority of candidates identified and fixed the problem with the array (starting at 0 rather than the 1 given). Fewer candidates were able to identify and fix the issue relating to the total being set to zero for each iteration.

A number of candidates identified this second issue but were either not precise enough with their response (suggesting that the line could be moved but not stating where it should be moved to) or did not attempt to go beyond the simple identification.

Candidates should be clear that this question required detail of how the program could be refined to fix problems, not just to identify problem(s).

Question 5 (e)

(e) The hotel car park charges £4 per hour. If the car is electric, this price is halved to £2 per hour.

Write an algorithm to:

- take as input the number of hours the user has parked and whether their car is electric or not
- calculate and output the total price
- repeat continually until the user enters 0 hours.

You must use **either**:

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

....

....

....

....

....

.... **[6]**

This question was relatively well answered by candidates.

Candidates were generally able to create suitable high-level program code to calculate and output the total price based on the information given.

Many candidates ignored the requirement to repeat until 0 was entered; in this case, 4 out of the 6 marks were still available.

To achieve marks for iteration it needed to both repeat the correct parts of the program and correctly terminate as per the requirements given.

A typical mistake was to repeatedly calculate the price but not ask afresh for new inputs.

Supporting you

Post-results services

If any of your students' results are not as expected, you may wish to consider one of our post-results services. For full information about the options available visit the OCR website.

Keep up-to-date

We send a weekly roundup to tell you about important updates. You can also sign up for your subject specific updates. If you haven't already, sign up here.

OCR Professional Development

Attend one of our popular CPD courses to hear directly from a senior assessor or drop in to a Q&A session. Most of our courses are delivered live via an online platform, so you can attend from any location.

Please find details for all our courses on the relevant subject page on our website or visit OCR professional development.

Signed up for ExamBuilder?

ExamBuilder is the question builder platform for a range of our GCSE, A Level, Cambridge Nationals and Cambridge Technicals qualifications. Find out more.

ExamBuilder is **free for all OCR centres** with an Interchange account and gives you unlimited users per centre. We need an Interchange username to validate the identity of your centre's first user account for ExamBuilder.

If you do not have an Interchange account please contact your centre administrator (usually the Exams Officer) to request a username, or nominate an existing Interchange user in your department.

Active Results

Review students' exam performance with our free online results analysis tool. It is available for all GCSEs, AS and A Levels and Cambridge Nationals.

It allows you to:

- review and run analysis reports on exam performance
- analyse results at question and/or topic level
- compare your centre with OCR national averages
- identify trends across the centre
- facilitate effective planning and delivery of courses
- identify areas of the curriculum where students excel or struggle
- help pinpoint strengths and weaknesses of students and teaching departments.

Find out more.

Need to get in touch?

If you ever have any questions about OCR qualifications or services (including administration, logistics and teaching) please feel free to get in touch with our customer support centre.

Call us on
01223 553998

Alternatively, you can email us on
support@ocr.org.uk

For more information visit
- **ocr.org.uk/qualifications/resource-finder**
- **ocr.org.uk**
- **/ocrexams**
- **/ocrexams**
- **/company/ocr**
- **/ocrexams**

We really value your feedback

Click to send us an autogenerated email about this resource. Add comments if you want to. Let us know how we can improve this resource or what else you need. Your email address will not be used or shared for any marketing purposes.

I like this **I dislike this**

Please note – web links are correct at date of publication but other websites may change over time. If you have any problems with a link you may want to navigate to that organisation's website for a direct search.

CAMBRIDGE
UNIVERSITY PRESS & ASSESSMENT