Qualification Accredited

Oxford Cambridge and RSA

GCSE (9-1)

Examiners' report

COMPUTER SCIENCE

J277

For first teaching in 2020

J277/02 Summer 2024 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 responses is 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.

Paper 2 series overview

This is the third examination series for the 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 responses using either OCR Exam Reference Language or a high-level language that they have studied.

Where candidates had extensive practise 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. In particular, identifying logic errors, functions, selection and iteration are all covered in the J277 specification and repeated experience of these in a practical setting helped candidates to be able to confidently answer questions in this paper.

Centres are encouraged to be aware of the contents of the specification, particularly Section 3c which shows details of OCR Exam Reference Language. Although candidates do not have to use this in their responses, questions will be presented in this way and candidates should be aware of these conventions in order for them to successfully understand and access examination questions.

Candidates who did well on this paper generally:	**Candidates who did less well on this paper generally:**
• were able to decompose a problem given and attempt each section, meaning that marks were able to be given for elements of each question even if the overall solution was not completely correct • understood the differences between interpreted and compiled code and why high and low-level languages are appropriate in certain scenarios • were able to use parameters passed into a function without asking for additional inputs • demonstrated an understanding of abstraction and decomposition in Section B.	• made simplistic errors in algorithms, such as including spaces in variable names or using incorrect comparisons • were vague in responses, typically giving examples in place of specific tightly defined definitions • gave generic responses where context was given • confused maintainability of a program with testing.

Section A overview

Section A consists of multiple questions and scenarios. Where algorithms are asked for, 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.

Section A begins with a number of simple, low challenge questions in order to make sure candidates are eased into the examination; this is deliberate by design for the J277 specification.

Question 1

1 Tick (✓) **one** box in each row to identify the programming construct where each keyword is used.

Keyword	**Programming construct**	
	Selection	**Iteration**
`if`		
`for`		
`while`		

[3]

This question was answered very well by the majority of candidates, showing a good understanding of the difference between the key programming constructs of selection and iteration.

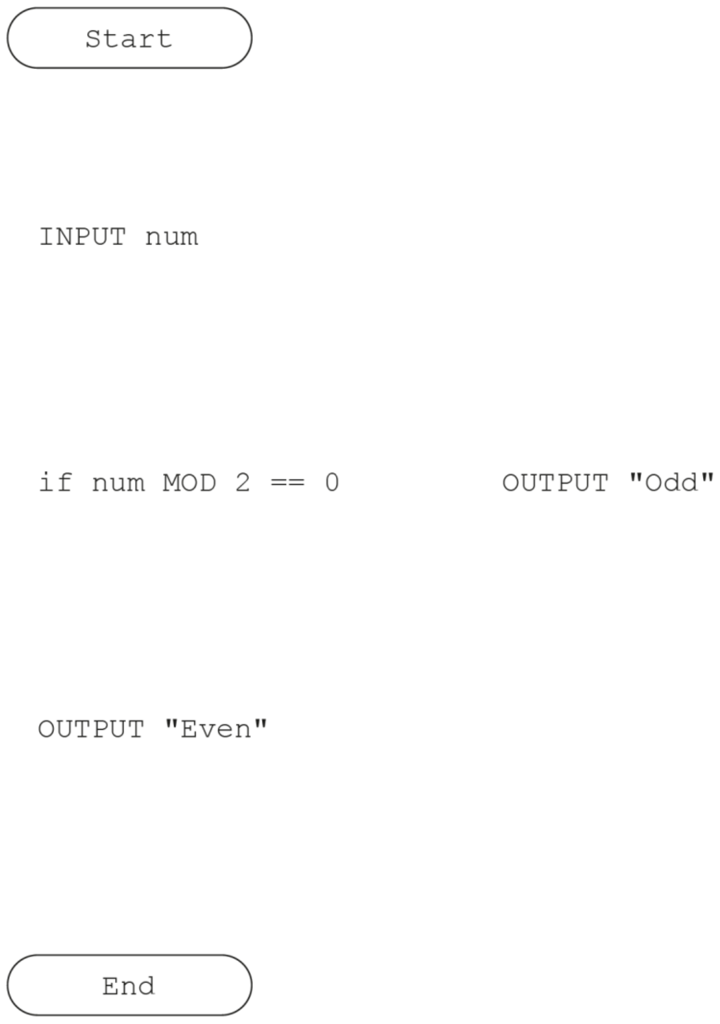
Practical programming skills in lessons should be used to make this explicit link between technical definitions and use within a high-level language.

Question 2

2 An algorithm decides if a number is odd or even.
An odd number divided by 2 will give the remainder 1.

The flowchart statements have been written for the algorithm, but the flowchart is incomplete.

Complete the flowchart.



[4]

The majority of candidates recognised the correct flowchart shapes to be used for a decision and many also were able to use the parallelogram shape for inputs and outputs. However, fewer correctly connected up the boxes to make sure that all paths started and ended at appropriate points and even fewer candidates labelled up the decision box appropriately with True/False or Yes/No. These labels are crucial to be able to follow the path of the algorithm when a decision is made.

Question 3 (a)

3
(a) State what is meant by the term syntax error. Give **one** example of a syntax error in a program.

Definition ....

....

Example ....

....
[2]

This question firstly asked for a definition of the term 'syntax error'. Although many candidates were correctly able to do this in terms of rules of the programming language being broken, many instead gave examples of issues that would cause syntax errors, such as 'where a bracket is missing'. This would indeed cause a syntax error in many high-level languages but is not a definition in general and so was not credited by examiners for this part of the question.

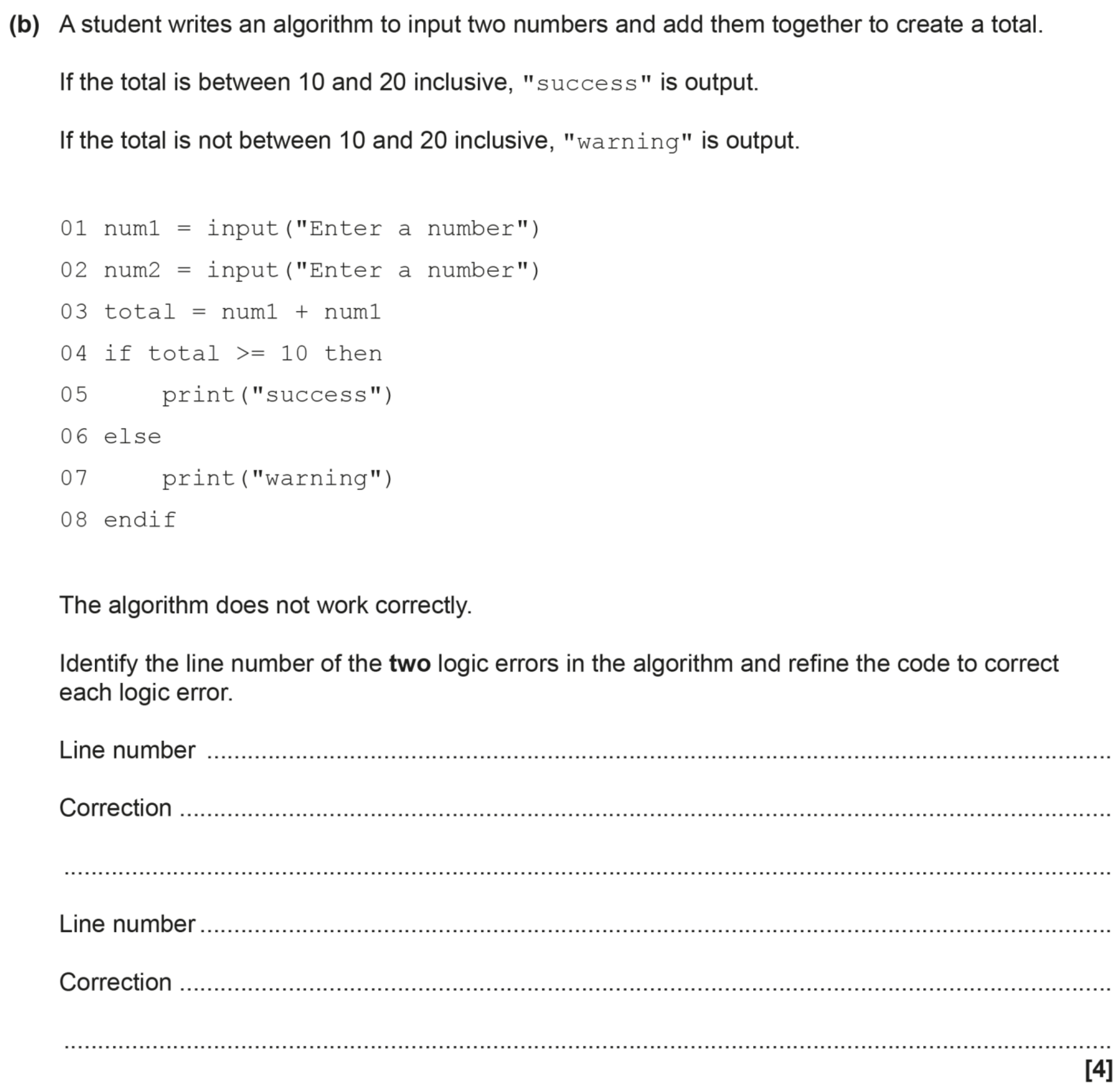
The second part did ask for an example and generous interpretation was asked of examiners so that any issue that could feasibly cause a syntax error was awarded, such as missing brackets or misspelling of key words. One common misconception here involved missing quotation marks/string delimiters around a string, which could instead be a reference to a variable if this was a single word.

Misconception

A syntax error is a mistake with the rules/grammar of the programming language that means the program will not run/execute or compile.

Code such as `print(temp)` would not necessarily be a syntax error because `temp` could plausibly be a variable.

Question 3 (b)



(b) A student writes an algorithm to input two numbers and add them together to create a total.

If the total is between 10 and 20 inclusive, `"success"` is output.

If the total is not between 10 and 20 inclusive, `"warning"` is output.

```
01 num1 = input("Enter a number")
02 num2 = input("Enter a number")
03 total = num1 + num1
04 if total >= 10 then
05     print("success")
06 else
07     print("warning")
08 endif
```

The algorithm does not work correctly.

Identify the line number of the **two** logic errors in the algorithm and refine the code to correct each logic error.

Line number ....

Correction ....

....

Line number ....

Correction ....

....

[4]

Line 03 was commonly identified as containing a logic error and many candidates were able to correct this. Where a candidate attempted to explain what changes should be made, this was only credited where the explanation was unambiguous and precise.

The correction to line 04 was commonly done incorrectly due to the need to check multiple values.

Misconception

Where multiple values are required to be checked in a selection statement, a line such as :

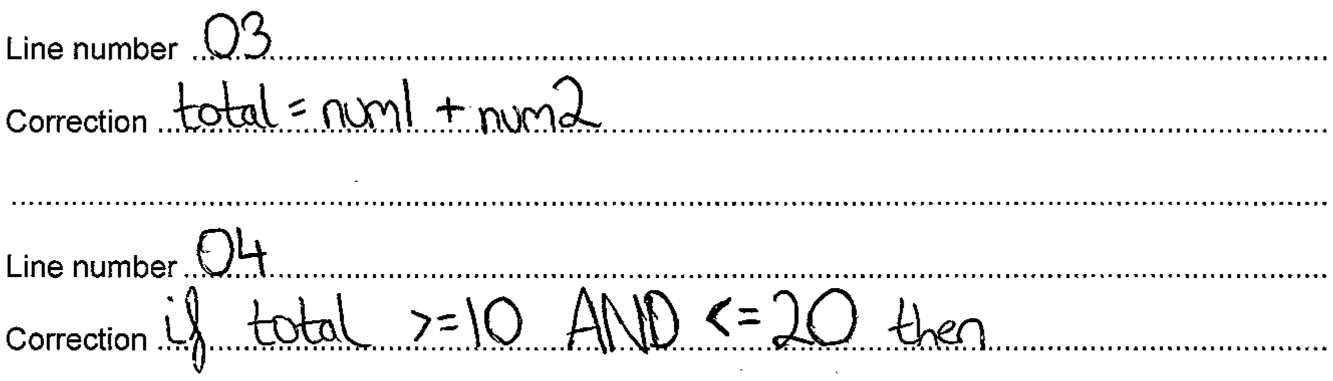
```
if total >= 10 and <= 20 then
```

is incorrect as the second part of the statement has nothing to compare 20 against. The first part will clearly evaluate to `true` or `false`, but the second part is ambiguous. Instead, candidates should be encouraged to use :

```
if total >= 10 and total <= 20 then
```

Alternatives that work in high-level languages would also obviously be accepted.

Exemplar 1



Line number 03

Correction total = num1 + num2

Line number 04

Correction if total >=10 AND <=20 then

Although this candidate has correctly identified lines 03 and 04, the correction for line 04 is incorrect due to the missing reference to total when comparing the upper boundary. This achieves three marks out of four.

Question 4 (a)

4 A program allows users to search for and watch videos. Users give a rating to the videos they watch.

(a) Identify **one** input and **one** output for the program.

Input ....

Output ....

[2]

Identification of inputs and outputs for a problem is covered in specification point 2.1.2 and candidates are expected to be precise with their responses. Answers such as 'video' as input are problematic because the candidate would not upload or use a video for their input. Candidates who were more precise and used terms such as 'name of video' of 'keywords searched for' were positively recognised for this.

Question 4 (b)

(b) Describe **one** method of defensive design that can be used when creating the program.

[2]

The specification (Section 2.3.1) lists multiple ways that defensive design could be used in a program and any of these, plus other sensible options, were allowed as an acceptable response. The describe command word then required candidates to add further detail to obtain a second mark, in this case by describing how it could be used, either generically or as a specific example. Many points on the mark scheme crossed over with each other, such as a validation example being a sensible expansion for anticipating misuse, and hence two marks were able to be obtained in multiple ways.

Question 5 (b)

(b) Draw a logic circuit for **P = NOT A AND (B OR C)**

A

B

P

C

[3]

Drawing logic circuits is now a commonly asked question and candidates are generally competent at doing this. Where issues did arise, they tended to be missing the circle from the NOT gate (and therefore turning it into a buffer, not a logic gate) or including the incorrect number of inputs into a gate.

Key point – logic gates

Candidates are expected to be able to draw the correct shapes for each gate. A number of candidates labelled their gates up, but this is not necessary; examiners are instructed to mark the shape of each gate and ignore any labelling.

Candidates are **not** allowed to take stencils or other tools that allow them to more easily draw these gates into an examination unless as part of a specific access arrangement agreed by OCR.

Question 6 (a)

6 The variable `message` is assigned a value.

```
message = "abcd1234"
```

(a) Complete the table to show the output when each statement executes.

The first output has been completed for you.

Statement	**Output**
`print(message.length)`	8
`print(message.upper)`	
`print(message.left(4))`	
`print(int(message.right(4))*2)`	

[3]

This question assessed string manipulation and used OCR Exam Reference Language(ERL) to present each statement. This highlights the importance of candidates being taught how to interpret ERL as questions in the exam will be written in this format.

Candidates were generally confident with the use of .upper and .left(). Candidates were less confident when this was combined with casting in the last part.

Question 6 (b)

(b) Write an algorithm in pseudocode to:

- store `"Hello"` in the variable `word1`
- store `"Everyone"` in the variable `word2`
- concatenate `word1` and `word2` to store `"HelloEveryone"` in the variable `message`

....

....

....

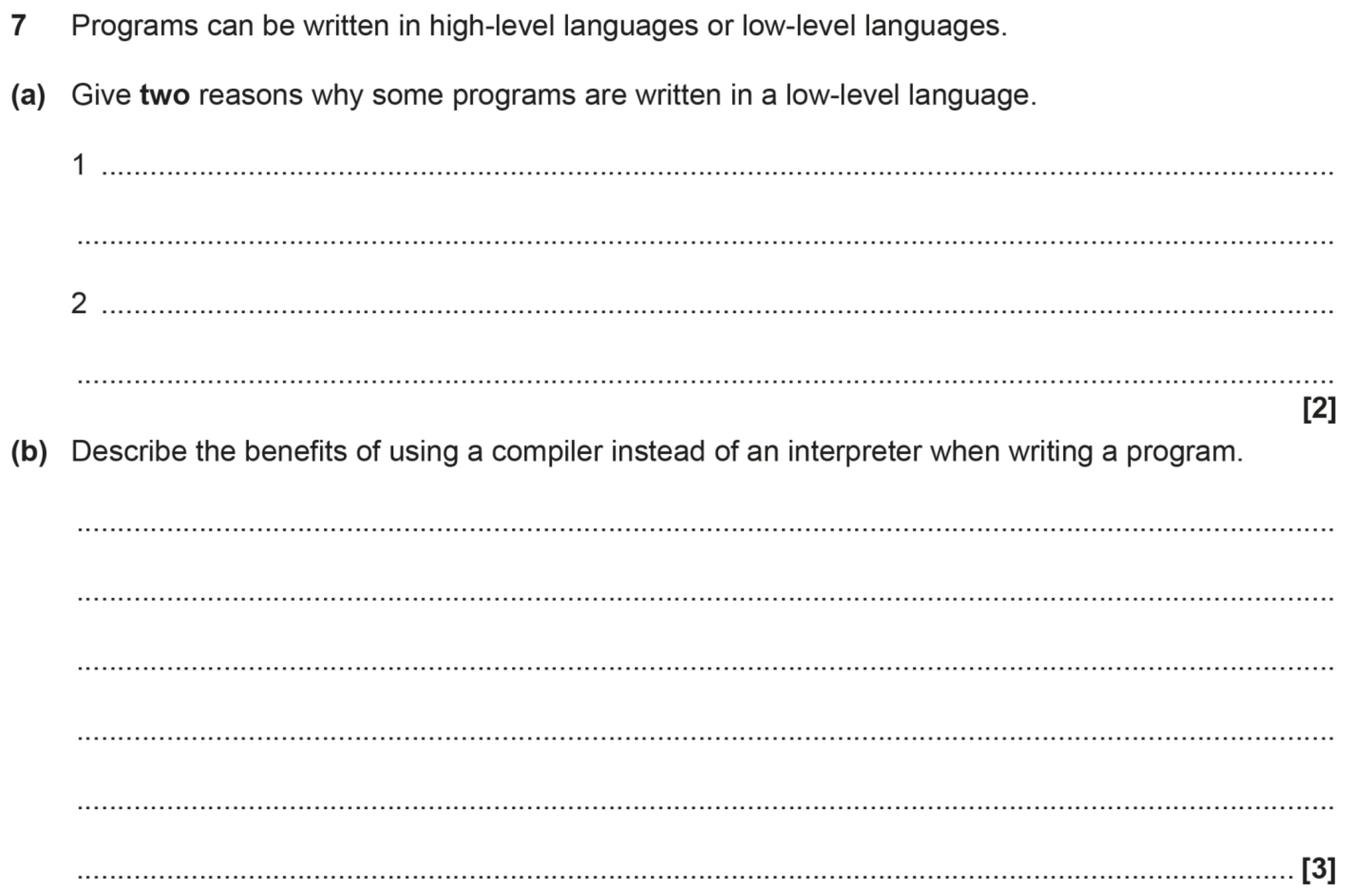
.... **[3]**

This question was intended to be straightforward and aimed at all candidates, including those expecting to achieve lower grades. However, a significant number of candidates dropped marks, either through perhaps not reading the question requirements or through not understanding the term concatenation.

There is no requirement in the question to print out the message obtained, simply to store it in the variable `message`. Where this was done in addition, this was not penalised but often it was done instead of storing the concatenated message.

A common mistake was to use the comma for concatenation when in most high-level languages (and especially Python), this is simply used to print out two items or pass two arguments to a subroutine and not actually concatenate values. Another common mistake was with the names of the variables given – if these included spaces or differed from those given in the question (such as `word 1`, `wordone` or `world1` instead of `word1`) examiners were instructed to not allow these alternatives.

Question 7 (a) and (b)



7 Programs can be written in high-level languages or low-level languages.

(a) Give **two** reasons why some programs are written in a low-level language.

1 ....

....

2 ....

.... **[2]**

(b) Describe the benefits of using a compiler instead of an interpreter when writing a program.

....

....

....

....

....

.... **[3]**

These questions assessed understanding of low-level languages and the use of compilers to translate from high-level code. Many candidates were able to do this well and clearly understood the need for high-level languages to be translated into machine code before the processor can execute this. The questions, however, did not ask just for this but specifically asked for reasons and benefits. Where candidates left their response as simply a discussion of the difference between high-level and low-level code or between translators and compilers, it was difficult for examiners to award marks.

Better responses were able to give clear reasons for the use of low-level code (such as direct control of hardware and faster execution of code) and the benefits of using a compiler instead of an interpreter (such as being able to distribute an executable file with no access to source code and not needing to translate code again once this has been done).

Question 8 (a)

8 An algorithm stores the position of a character on a straight line as an integer. A user can move the character left or right.

The following algorithm:

- generates one random number between 1 and 512 (inclusive) to store as the position
- prompts the user to input a direction to move (left or right)
- takes a direction as input until a valid direction is input.

```
p = random(1, 512)
print("The position is ", p)
a = ""
while a != "left" and a != "right"
     a = input("Enter direction, left or right")
endwhile
```

(a) Describe **two** ways to improve the maintainability of the algorithm.

1 ....

....

....

2 ....

....

.... **[4]**

It was pleasing that the majority of candidates understood the term maintainability and were able to suggest suitable improvements on a generic level, such as improving the naming of variables and adding comments. Better responses that achieved full marks were able to apply this to the code given, such as suggesting more suitable variable names.

Candidates who suggested adding indentation were not credited with marks as this has clearly already been done in the code given and thus would not be an improvement.

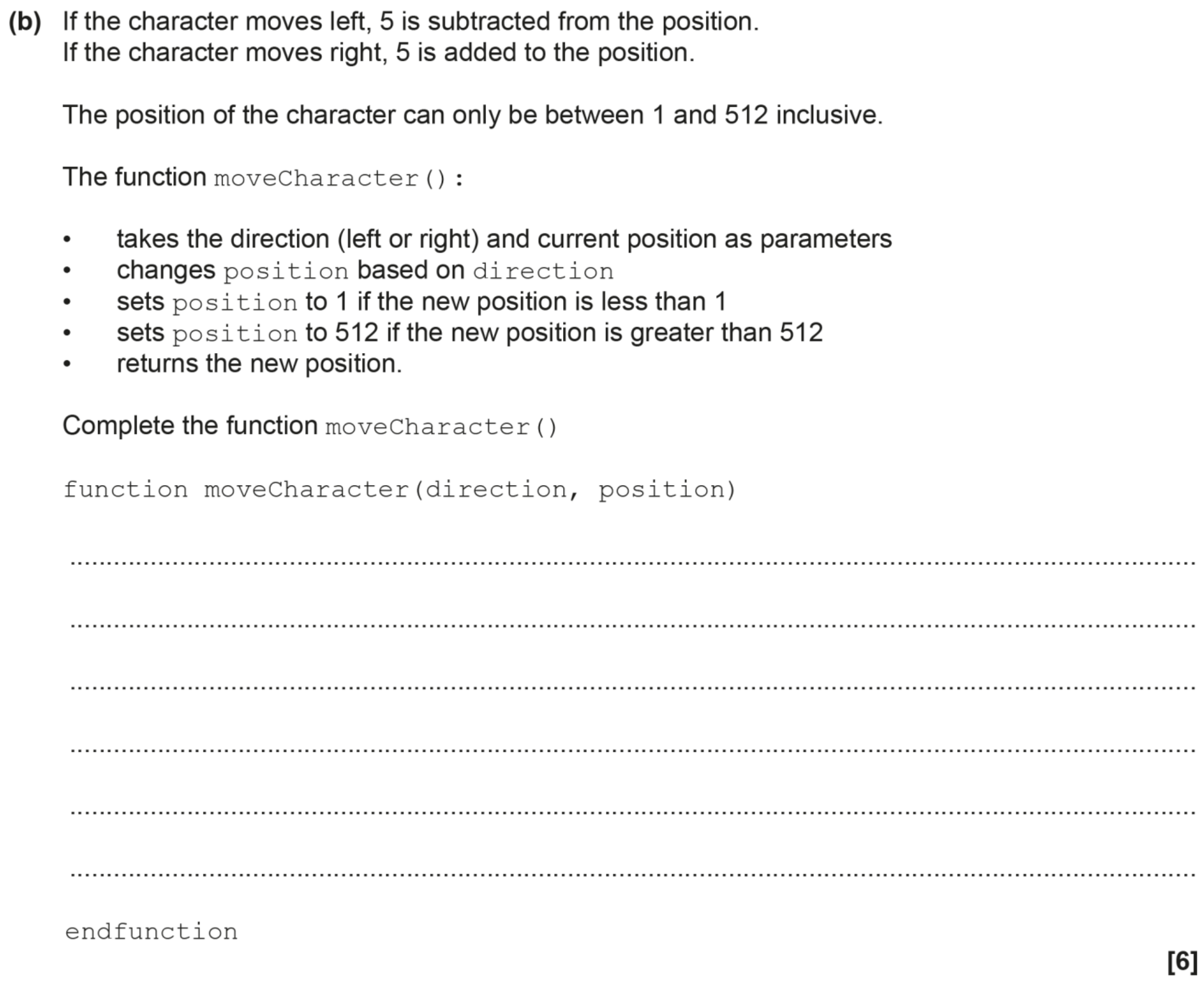
Key point – generic versus. context driven responses

Where a question gives a context or scenario (such as data or a program being given), it is important that candidates refer to this context in their response if they are hoping to achieve full marks. For this question, the question text was:

'Describe two ways to improve the maintainability of **this** algorithm'

This is a very different question from 'Describe two ways to improve the maintainability of **an** algorithm' where a generic response would suffice.

Question 8 (b)



(b) If the character moves left, 5 is subtracted from the position.
If the character moves right, 5 is added to the position.

The position of the character can only be between 1 and 512 inclusive.

The function `moveCharacter()`:

- takes the direction (left or right) and current position as parameters
- changes `position` based on `direction`
- sets `position` to 1 if the new position is less than 1
- sets `position` to 512 if the new position is greater than 512
- returns the new position.

Complete the function `moveCharacter()`

```
function moveCharacter(direction, position)

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

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

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

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

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

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

endfunction
```

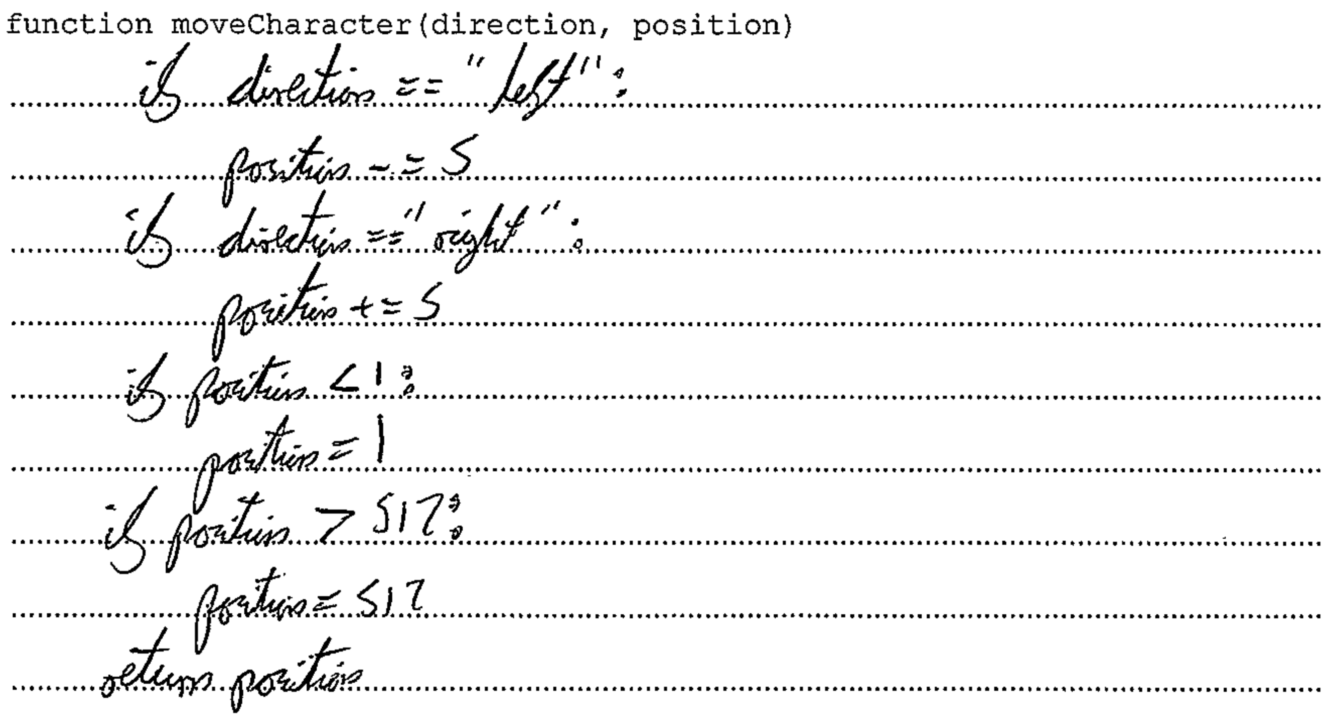
[6]

This question was an excellent discriminator in terms of candidate achievement. In particular, correct use of the given parameters (`direction` and `position`) was only seen from the most successful candidates. Many candidates instead started their response by asking for input from the user and therefore overwriting the parameters, losing crucial marks in the process.

As on previous papers, this question provided one mark for any attempt made at a section of the requirement, in this case selection. Candidates who attempted to use an `if` or `select case` statement, even incorrectly or in the wrong context were able to achieve at least this mark; centres should therefore continue to encourage all candidates to attempt each and every question and not leave responses blank.

It was challenging to achieve full marks, but many candidates did because of their excellent levels of practical programming experience in school.

Exemplar 2



```
function moveCharacter(direction, position)
    if direction == "left":
        position -= 5
    if direction == "right":
        position += 5
    if position < 1:
        position = 1
    if position > 512:
        position = 512
    return position
```

This candidate achieved full marks. The given parameters (direction and position) are both used and not overwritten by inputs before the position is modified depending on the direction given. The position is then validated to make sure that it is between 1 and 512 before being returned.

Note the use of `position +=5` to add 5 to the position. This is an entirely acceptable alternative to `position = position + 5`

Section B overview

Section B consists of multiple sub-questions all centred around one scenario (in this case, a school sports day). 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 are presented using flowcharts or structured English in Section B, no marks were 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 using and so responses are marked for their logical correctness and consistency. This includes where responses combine features from multiple high-level languages.

Question 9 (a) (i)

9 Students take part in a sports day. The students are put into teams.

Students gain points depending on their result and the year group they are in. The points are added to the team score.

The team with the most points at the end of the sports day wins.

(a) Data about the teams and students is stored in a sports day program.

(i) Identify the most appropriate data type for each variable used by the program.

Each data type must be different.

Variable	Example	Data type
`teamName`	`"Super-Team"`	
`studentYearGroup`	`11`	
`javelinThrow`	`18.2`	

[3]

This question was completed very well by the vast majority of candidates, showing that the use of data types are now well understood by centres.

Question 9 (a) (ii)

(ii) The student names for a team are stored in an array with the identifier `theTeam`

An example of the data in this array is shown:

Index	0	1	2	3	4	5
Data	Ali	Eve	Ling	Nina	Sarah	Tom

`theTeam`

A linear search function is used to find whether a student is in the team. The function:

- takes a student name as a parameter
- returns `True` if the student name is in the array
- returns `False` if the student name is **not** in the array.

Complete the design of an algorithm for the linear search function.

```
function linearSearch(studentName)

    for count = 0 to ....................................

        if theTeam[....................................] == .................................... then

            return ....................................

        endif

    next count

    return False

endfunction
```

[4]

This was a challenging question revolving around the use of an array that is iterated through to implement a linear search. Many candidates achieved two marks for the first and last points, understanding that the loop would repeat from indexes 0 to 5 and that the value `True` would be returned if the item was found. As usual, allowance was given for off-by-one errors with this loop because of the prevalence of Python in centres and how loops are count controlled loops are handled in this language.

It was far less common for candidates to achieve the middle two marks, perhaps because the level of technical knowledge needed was greater. A number of candidates attempted here to fashion a 2D array and refer to multiple indexes, but this was not appropriate given the data structure given. The most challenging mark was certainly the use of `count` as the index of the `theTeam` array, with very few candidates correctly identifying this as the missing element.

Assessment for learning

Centres are encouraged to link the topics of arrays (both single and two-dimensional) to count controlled loops to be confident in answering questions like this one.

A very common programming exercise is to iterate through every item in an array, either to search for an item, count or add items or as the pre-cursor for a search. Candidates are expected to have significant practical programming experience over the duration of their studies.

Question 9 (b)

(b) This algorithm calculates the number of points a student gets for the distance they throw in the javelin:

```
01   javelinThrow = input("Enter distance")
02   yearGroup = input("Enter year group")
03   if javelinThrow >= 20.0 then
04       score = 3
05   elseif javelinThrow >= 10.0 then
06       score = 2
07   else
08       score = 1
09   endif
10   if yearGroup != 11 then
11       score = score * 2
12   endif
13   print("The score is", score)
```

Complete the trace table for the algorithm when a student in year 10 throws a distance of 14.3

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

Line number	`javelinThrow`	`yearGroup`	`score`	**Output**

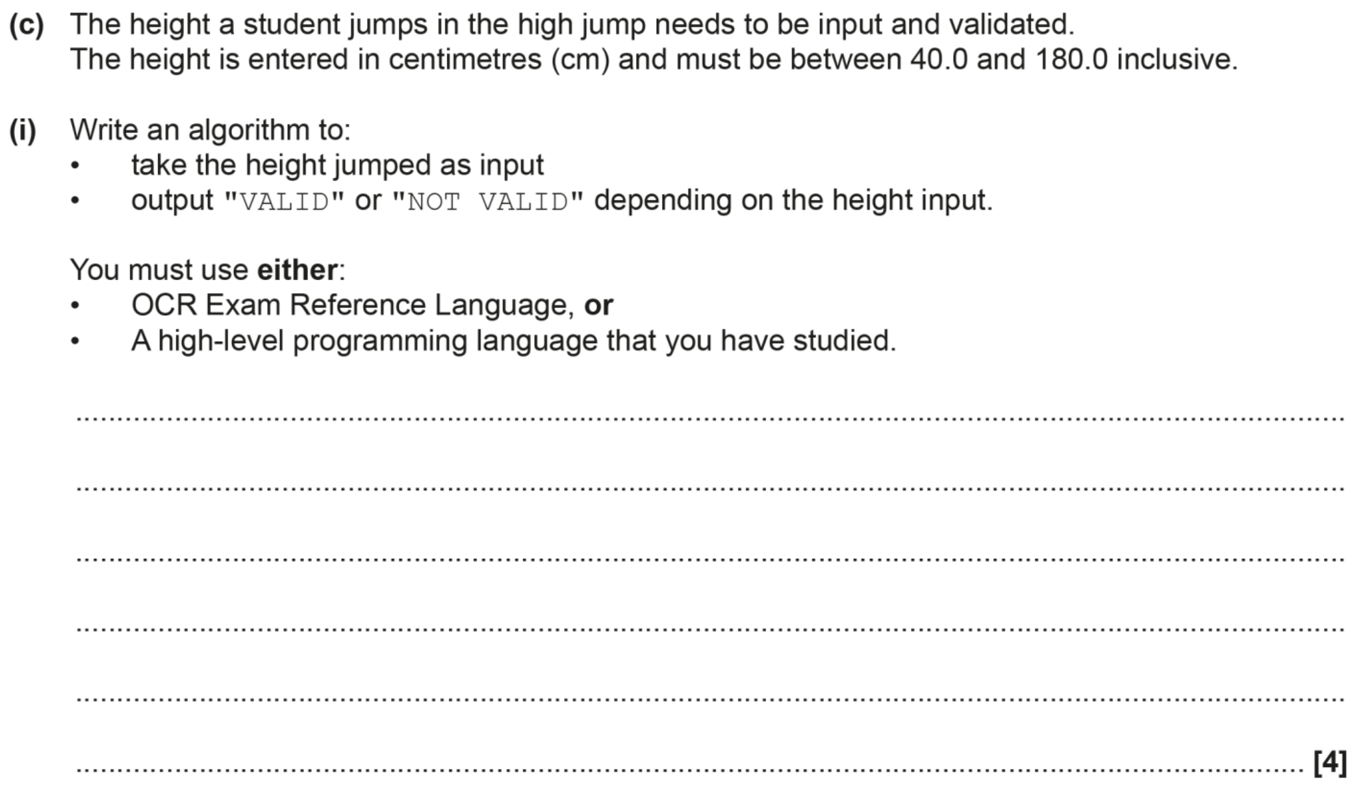
[4]

Trace tables have appeared multiple times in J277 examination papers now and candidates are hopefully familiar with the expectations, which are consistent across series.

Most candidates correctly traced through the given algorithm, which was more accessible than usual due to not containing any loops. Where mistakes were made, this was typically to do with either incorrect line numbers being given for each change (this was penalised once only and then subsequent mistakes with line numbers followed through) or additional incorrect output being given.

Candidates should be encouraged to simply leave boxes blank if no output is given on a particular line. If (for example) ‘x’ is written, examiners are unsure whether the candidate meant no output or the letter ‘x’ should be output. This ambiguity would mean that no mark could be given.

Question 9 (c) (i)



(c) The height a student jumps in the high jump needs to be input and validated.
The height is entered in centimetres (cm) and must be between 40.0 and 180.0 inclusive.

(i) Write an algorithm to:

- take the height jumped as input
- output `"VALID"` or `"NOT VALID"` depending on the height input.

You must use **either**:

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

[4]

This question was done very well by the majority of candidates, with multiple ways of achieving the marks possible.

One common problem was consistent with Question 3 (b), as again multiple conditions could be evaluated using a single `if` statement. Candidates needed to refer to their input value for each comparison if they were to achieve full marks. Another common problem was with the boundaries used; the tests must check 40 to 80 inclusive (for valid response) to be marked as correct. Obviously, if an input on these boundaries would produce the wrong output then full marks could not be given.

One final common problem involved the use of greater than or equal to signs (and also less than or equal to). The common signs used in mathematics are not available on a typical keyboard and so would not be allowed in Section B of this paper. Instead, **>=** or **<=** should be used, and these should be the correct way around.

Question 9 (d)

(d) The individual results for each student in each event are stored in a database.

The database table `TblResult` stores the times of students in the 100 m race. Some of the data is shown:

StudentID	YearGroup	TeamName	Time
11GC1	11	Valiants	20.3
10VE1	10	Super-Team	19.7
10SM1	10	Super-Team	19.2
11JP2	11	Champions	19.65

Complete the SQL statement to show the Student ID and team name of all students who are in year group 11

```
SELECT StudentID, ..............................................

FROM ...........................................................................

...................................................................................
```

[4]

A number of candidates struggled with this question. Problems included spaces in field names, misspelling of the table name (such as `TblResult`**s** plural when `TblResult` singular was given) or misunderstanding of the WHERE clause.

Allowance was given where == was used for comparison and examiners were instructed to allow this (as this is used for comparison in high-level languages such as Python), although this is incorrect as defined in the most recent ANSI SQL standards.

Question 9 (e) (i) and e (ii)

(e) Abstraction and decomposition have been used in the design of the sports day program.

(i) Identify **one** way that abstraction has been used in the design of this program.

....

.... **[1]**

(ii) Identify **one** way that decomposition has been used in the design of this program.

....

.... **[1]**

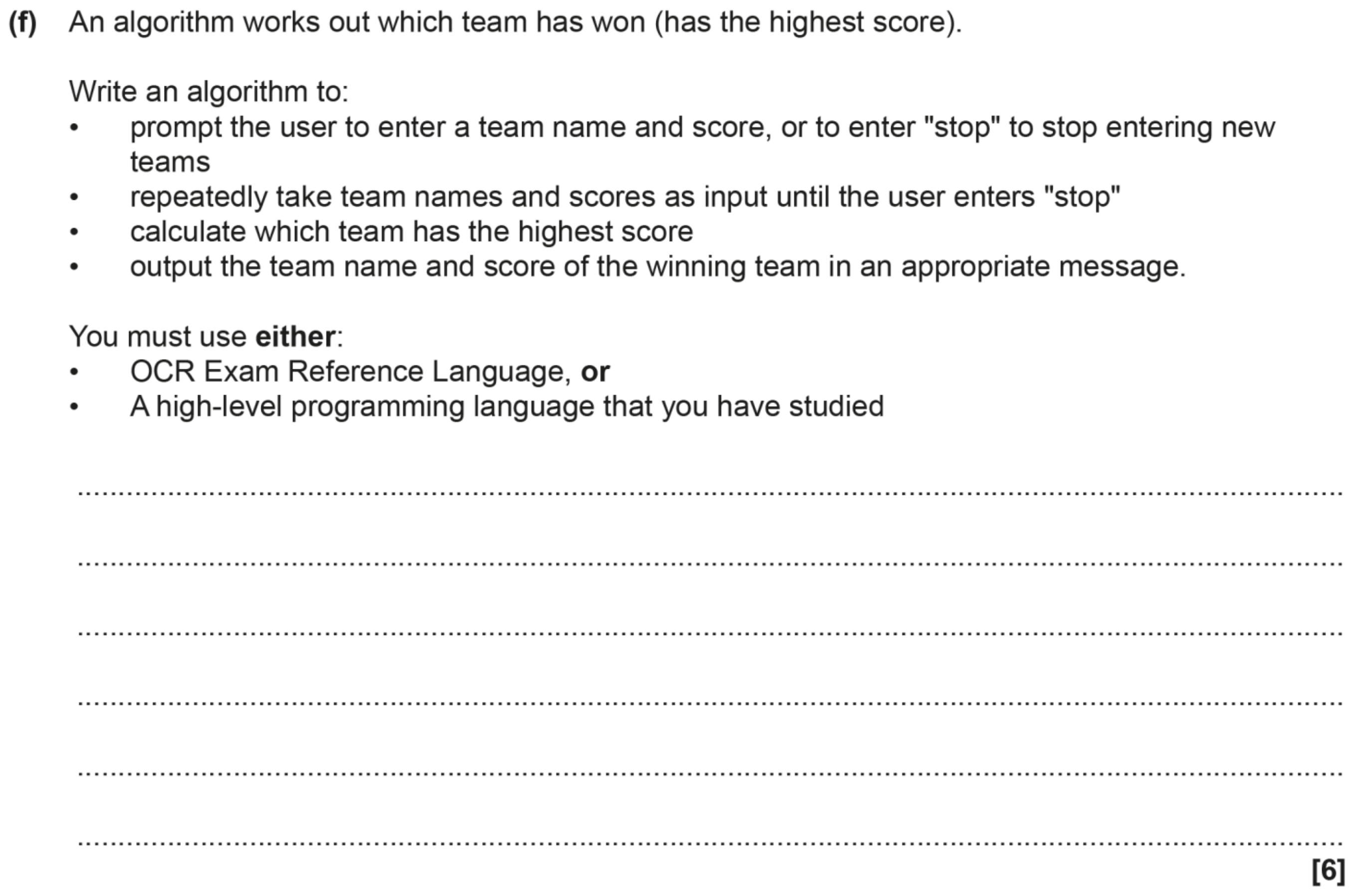
Both questions here asked about abstraction and decomposition of the sports day program. As explained previously in this report, where a scenario or context is given, candidates are expected to use this context. No marks were given by examiners for generic definitions of what the term abstraction or decomposition means.

Abstraction in the sports day program could have been for focusing on anything sensible (such as event names) or removing/ignoring anything sensible (such as showing student IDs instead of names). Where the context of the sports day was used, candidates were generally successful in achieving this mark.

Decomposition use was more tricky to correctly identify, as many candidates simply referred to how already separate data was stored. Where this extended to true decomposition (such as breaking down data into multiple tables, splitting up event data, etc.) this was credited but the average candidate fell short here. A much more successful approach was to discuss the decomposition of the program, such as having a separate algorithm for each event. Candidates attempting this angle of response did very well.

Examiners were instructed for both questions to be generous in deciding whether candidates had indeed referred to the sports day context.

Question 9 (f)



(f) An algorithm works out which team has won (has the highest score).

Write an algorithm to:

- prompt the user to enter a team name and score, or to enter "stop" to stop entering new teams
- repeatedly take team names and scores as input until the user enters "stop"
- calculate which team has the highest score
- output the team name and score of the winning team in an appropriate message.

You must use **either**:

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

[6]

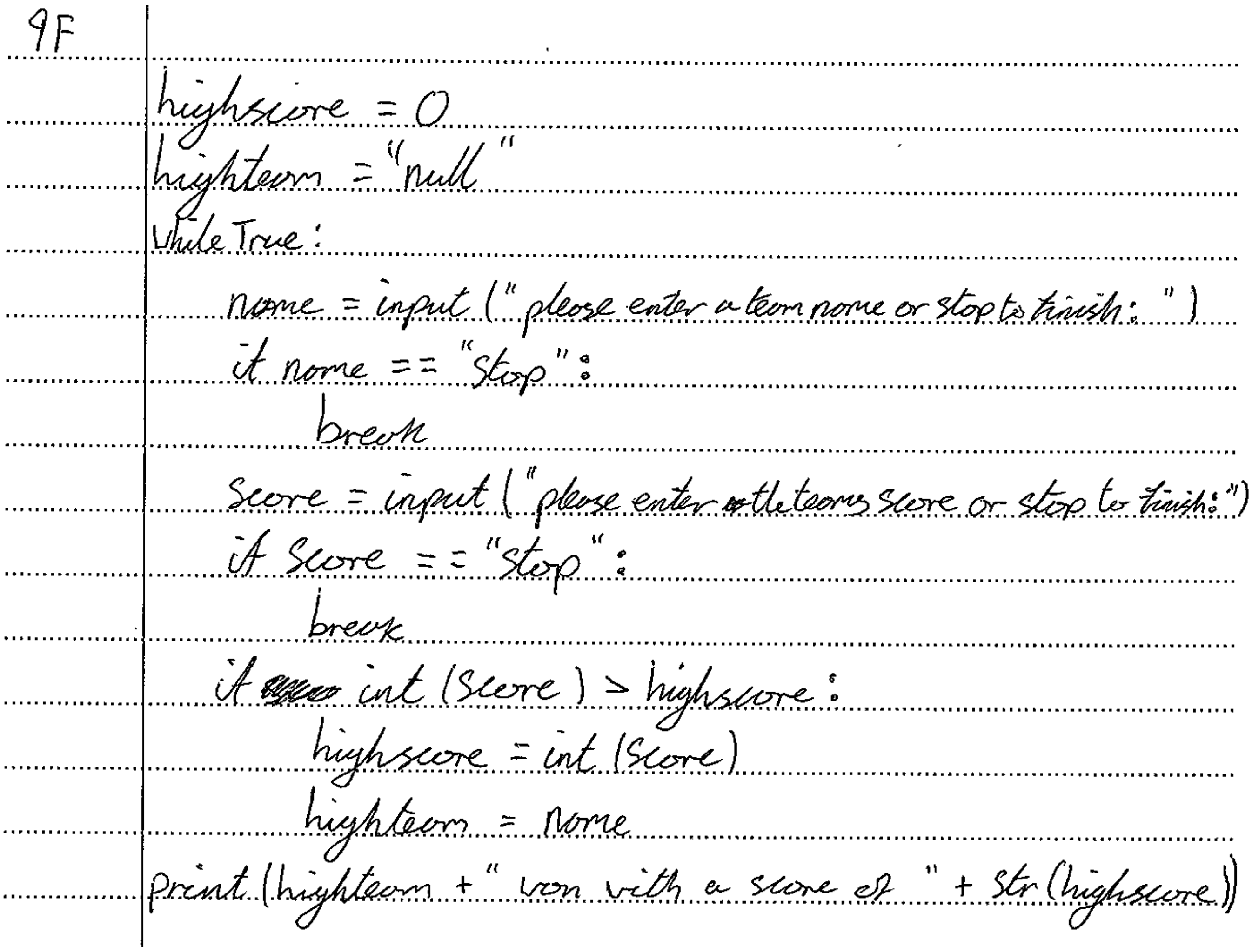
The final question in Section B is expected to be challenging and this proved to be the case, although again was perhaps more accessible than previous papers' final questions.

Marks were available for inputs (one mark) and correctly iterating over as required (two marks), with these three marks proving to be the easiest to achieve. The next three marks required significant processing in terms of calculating the highest score and team name from multiple values entered by the user. The vast majority of candidates simply kept a running 'highest score' and updated this on each iteration. Where this was attempted, it was mostly successful. Other candidates attempted more complex solutions, including adding data to arrays and then calculating highest values; where this was done successfully, this of course achieved full marks but frequently small logic mistakes meant that not all marks were given. Centres should encourage candidates to keep their responses simple and not produce over-elaborate solutions if a simpler alternative is available.

A common mistake was where candidates attempted to use loops in the style of `for x in list:`. In this case, the variable `x` is a reference to an item in the array and not an index. It would therefore not be appropriate to try to access `list[x]` later in the code.

A significant number of candidates were able to create a solution that fully met the requirements of the question, doing so in an elegant and efficient manner. This is extremely pleasing and show excellent understanding, produced from excellent teaching and significant amounts of practical programming experience.

Exemplar 3



9F

```
highscore = 0
highteam = "null"
While True:
    name = input("please enter a team name or stop to finish: ")
    if name == "stop":
        break
    Score = input("please enter the teams score or stop to finish:")
    if Score == "stop":
        break
    if int(Score) > highscore:
        highscore = int(Score)
        highteam = name
print(highteam + " won with a score of " + str(highscore))
```

The candidate response shown here achieved six out of six marks. Both the `name` and `score` are input as required, with this being inside a while loop. Perhaps unconventionally (but acceptably), the candidate has used a `break` command to end the loop (which otherwise is infinite) upon stop being entered. This could have been more elegantly rewritten as `while name != 'stop'` but this would not have achieved any further marks.

Within the loop, the candidate uses two variables to keep track of the current highest score and associated team, before printing these out in a message once the loop has ended.

Supporting you

Teach Cambridge

Make sure you visit our secure website **Teach Cambridge** to find the full range of resources and support for the subjects you teach. This includes secure materials such as set assignments and exemplars, online and on-demand training.

Don't have access? If your school or college teaches any OCR qualifications, please contact your exams officer. You can forward them this link to help get you started.

Reviews of marking

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.

Access to Scripts

We've made it easier for Exams Officers to download copies of your candidates' completed papers or 'scripts'. Your centre can use these scripts to decide whether to request a review of marking and to support teaching and learning.

Our free, on-demand service, Access to Scripts is available via our single sign-on service, My Cambridge. Step-by-step instructions are on our website.

Keep up-to-date

We send a monthly bulletin 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 professional development 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 for your subject on **Teach Cambridge**. You'll also find links to our online courses on NEA marking and support.

Signed up for ExamBuilder?

ExamBuilder is a free test-building platform, providing unlimited users exclusively for staff at OCR centres with an Interchange account.

Choose from a large bank of questions to build personalised tests and custom mark schemes, with the option to add custom cover pages to simulate real examinations. You can also edit and download complete past papers.

Find out more.

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 (examined units only).

Find out more.

You will need an Interchange account to access our digital products. 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.

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
facebook.com/ocrexams
twitter.com/ocrexams
instagram.com/ocrexaminations
linkedin.com/company/ocr
youtube.com/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