

AQA style

GCSE Computer Science

Paper 1: Computational thinking and programming skills – Python (8525/1B) – Practice Paper

8525/1B · Practice paper

Time allowed: 2 hours

Candidate name	
Centre number	
Candidate number	

Materials

- There are no additional materials required for this paper.
- You must not use a calculator.
- You must not use a calculator.

Instructions

- Use black ink or black ball-point pen. Use pencil only for drawing.
- Answer all questions.
- You must answer the questions in the spaces provided.
- If you need extra space for your answer(s), use the lined pages at the end of this book. Write the question number against your answer(s).
- Do all rough work in this book. Cross through any work you do not want to be marked.
- Questions that require a coded solution must be answered in Python.
- You should assume that all indexing in code starts at 0 unless stated otherwise.

Information

- The total number of marks available for this paper is 90.
- AQA pseudo-code is used in some questions; you do not have to use it in your answers, but any pseudo-code you write must be clear, consistent and unambiguous.

Advice

- For the multiple-choice questions, completely fill in the lozenge alongside the appropriate answer.
- If you want to change your answer you must cross out your original answer.

Original practice material written by Teach io in the style of AQA GCSE Computer Science (8525). It is not produced, approved or endorsed by AQA.

Answer all questions.

Time allowed: 2 hours. Programming questions must be answered in Python.

01

01.1 Define the term abstraction.

[2 marks]

.....

.....

.....

01.2 Shade one lozenge to show which data type would be most suitable for storing the price of a cinema ticket, for example 8.75.

[1 mark]

A	Boolean	
B	Character	
C	Integer	
D	Real	

01.3 Shade one lozenge to show which data type would be most suitable for storing a customer's postcode, for example B74 4RD.

[1 mark]

A	Character	
B	Integer	
C	Real	
D	String	

01.4 The variable ref is assigned the value 'WX/2027-B'. Shade one lozenge to show the result of the AQA pseudo-code expression LEN(ref).

[1 mark]

A	7	
B	8	
C	9	
D	10	

01.5 Shade one lozenge to show the value of the AQA pseudo-code expression SUBSTRING(2, 5, 'KEYBOARD').

[1 mark]

A	'EYBO'	
B	'YBOA'	
C	'YBOAR'	
D	'KEYBO'	

[Total: 6 marks]

02

02.1 Shade one lozenge to show which type of iteration is definite (count-controlled).

[1 mark]

A	FOR ... ENDFOR	
B	REPEAT ... UNTIL	
C	WHILE ... ENDWHILE	
D	IF ... ENDIF	

02.2 Shade one lozenge to show the output of the following AQA pseudo-code:

```
FOR k ← 20 TO 4 STEP -4  
    OUTPUT k  
ENDFOR
```

[1 mark]

A	20 16 12 8	
B	20 16 12 8 4	
C	16 12 8 4	
D	4 8 12 16 20	

02.3 Shade one lozenge to show how many values are output by the following AQA pseudo-code:

```
n ← 5  
REPEAT  
    OUTPUT n  
    n ← n * 2  
UNTIL n > 60
```

[1 mark]

A	3	
B	4	
C	5	
D	6	

[Total: 3 marks]

03 A program calculates the VAT to be added to the price of items in a shop. The VAT rate is 20%.

Explain one reason why the programmer should store the VAT rate as a named constant rather than typing the value 0.2 each time it is used in the program.

[2 marks]

04 The Python program below processes a list of words.

```
words = ["sun", "river", "oak", "meadow"]
longest = ""
count = 0
for w in words:
    if len(w) > len(longest):
        longest = w
        count = count + 1
print(longest, count)
```

04.1 Complete the trace table for the program. You may not need to use all of the rows.

[3 marks]

w	longest	count	Output
	""	0	

04.2 State the output of the program if the first line was changed to:

words = ["ash", "elm", "yew"]

[1 mark]

[Total: 4 marks]

05 A flowchart describes an algorithm. [Diagram described in words: START → process box 'x ← 1, y ← 30' → decision diamond 'Is y > x?'. If Yes → process box 'x ← x * 2' → process box 'y ← y - 5' → back to the decision diamond. If No → output box 'OUTPUT x' → STOP.]

Complete the trace table for the algorithm shown in the flowchart. You may not need to use all of the rows.

[3 marks]

x	y	Output
1	30	

x	y	Output

06 The subroutine makeCode shown below is written in AQA pseudo-code. It creates a membership code from a person's surname and year of birth. Line numbers are shown but are not part of the algorithm.

```

1  SUBROUTINE makeCode(name, year)
2      code ← SUBSTRING(0, 2, name)
3      code ← code + INT_TO_STRING(year MOD 100)
4      RETURN code
5  ENDSUBROUTINE

```

06.1 State the value returned by the subroutine call makeCode('Patel', 2009). **[1 mark]**

06.2 State the identifier of the local variable used in the subroutine makeCode that is not a parameter. **[1 mark]**

06.3 The year part of the code should always be two digits. For example, makeCode('Patel', 2009) should return 'Pat09'.

Describe how the subroutine could be changed so that the year part is always two digits. **[2 marks]**

[Total: 4 marks]

07 A car park charges drivers as follows:

- up to and including 2 hours costs £2.50
- more than 2 hours and up to and including 5 hours costs £6.00
- more than 5 hours costs £10.00
- electric vehicles receive £1.00 off the charge.

Write a Python program that:

- asks the user to enter the number of hours they have parked
- asks the user to enter whether their vehicle is electric (Y or N)
- calculates the charge
- outputs the charge.

You should assume that the user enters a whole number of hours of 1 or more, and enters either Y or N.

You should use meaningful variable name(s), correct syntax and indentation in your answer.

[6 marks]

08 A merge sort is being used to sort the list [7, 2, 9, 4, 3, 8, 1, 6] into ascending order.

08.1 Complete the table to show how the merge sort combines the sub-lists into a single sorted list.

[3 marks]

Stage	Contents of sub-lists
After splitting into single items	[7] [2] [9] [4] [3] [8] [1] [6]
After first merge	
After second merge	
After final merge	

08.2 State one advantage of merge sort compared with bubble sort.

[1 mark]

.....

.....

[Total: 4 marks]

09 A record structure has been declared in AQA pseudo-code to store details about flights booked by a travel agent.

```

RECORD Flight
    flightNo : String
    seats : Integer
    fare : Real
    international : Boolean
ENDRECORD

f1 ← Flight('BA214', 3, 85.00, True)
f2 ← Flight('LM090', 2, 49.50, False)
trips ← [f1, f2]

```

09.1 Shade one lozenge to show the data type of the field international.

[1 mark]

A	Boolean	
B	Integer	
C	Real	
D	String	

09.2 Shade one lozenge to show the value of trips[0].seats.

[1 mark]

A	2	
B	3	
C	85.00	
D	'BA214'	

09.3 Shade one lozenge to show the value of the expression f1.seats * f1.fare.

[1 mark]

A	85.00	
B	170.00	
C	255.00	
D	340.00	

09.4 Write an AQA pseudo-code statement that changes the number of seats in f2 to 4. **[1 mark]**

[Total: 4 marks]

10 The algorithm below is written in AQA pseudo-code.

```

nums ← [5, 9, 6, 12, 4]
rises ← 0
biggest ← 0
FOR i ← 1 TO 4
    diff ← nums[i] - nums[i - 1]
    IF diff > 0 THEN
        rises ← rises + 1
        IF diff > biggest THEN
            biggest ← diff
        ENDIF
    ENDIF
ENDFOR
OUTPUT rises
OUTPUT biggest

```

10.1 Complete the trace table for the algorithm. You may not need to use all of the rows. **[5 marks]**

i	diff	rises	biggest	Output
		0	0	

10.2 Shade one lozenge to show the purpose of the algorithm's second output. **[1 mark]**

A	The largest value in the array	
----------	--------------------------------	--

B	The number of times a value is greater than the one before it	
C	The largest increase between two neighbouring values	
D	The total of all the increases	

[Total: 6 marks]

11 A youth club accepts members aged from 11 to 16 inclusive. A programmer has written the following Python code to validate a member's age.

```
age = int(input("Enter age: "))
if age > 10 or age <= 16:
    print("Age accepted")
else:
    print("Age rejected")
```

11.1 Complete the test table. For each test, give the type of test data, the expected result if the program were correct, and the actual result produced by the code shown.

[4 marks]

Test data	Type of test data	Expected result	Actual result
13		Accepted	Accepted
16	Boundary		
20			

11.2 Describe the change that needs to be made to the code so that it validates the age correctly.

[1 mark]

.....

.....

[Total: 5 marks]

12 The Python subroutine `binarySearch` below performs a binary search on a list of numbers in ascending order. It returns the index of target, or -1 if target is not found.

```
def binarySearch(nums, target):
    low = 0
    high = len(nums) - 1
    while low <= high:
        mid = (low + high) // 2
        if nums[mid] == target:
            return mid
        elif nums[mid] < target:
            low = mid + 1
        else:
            high = mid - 1
    return -1
```

12.1 The subroutine is called using `binarySearch([4, 9, 15, 21, 30, 38, 44, 57], 33)`.

Complete the trace table. The first row has been completed for you. You may not need to use all of the rows.

[4 marks]

low	high	mid
0	7	

12.2 Write a Python subroutine called `lastPosition` that:

- takes a list `items` and a value `target` as parameters
- uses iteration to check the items in the list
- returns the index of the last position where `target` is found
- returns `-1` if `target` is not in the list.

For example, `lastPosition([3, 8, 3, 5], 3)` should return `2`.

You must not use any built-in Python search functions or methods such as `index` or `rindex`, and you must not use the `in` operator to test whether `target` is in the list.

You should use meaningful variable name(s), correct syntax and indentation in your answer.

[7 marks]

12.3 Explain why the call `binarySearch([4, 9, 15, 21, 30, 38, 44, 57], 33)` returns -1.

[1 mark]

12.4 A sorted list contains 1000 items.

Compare the maximum number of comparisons needed by a linear search and by a binary search to find an item in this list.

[3 marks]

[Total: 15 marks]

13 A banking app asks a user to choose a 4-digit PIN. A valid PIN must be exactly 4 characters long and every character must be a digit (0–9), for example 0457.

A programmer has started to write a Python program to check PINs.

```
pin = input("Choose a PIN: ")
print("PIN set")
```

Extend the Python program so that it:

- checks the PIN is exactly 4 characters long
- checks every character of the PIN is a digit (0–9)
- outputs Invalid PIN and asks the user to enter a new PIN if it is not valid
- repeats until a valid PIN is entered
- outputs PIN set once a valid PIN is entered.

You should use meaningful variable name(s), correct syntax and indentation in your answer.

[6 marks]

14 The Python subroutine `applyDiscount` is shown below.

```
def applyDiscount(price, percent):  
    saving = price * percent / 100  
    newPrice = price - saving  
    return newPrice  
  
cost = applyDiscount(80, 15)  
print(cost)
```

14.1 State the value that is output when the program is run.

[1 mark]

14.2 Explain why `saving` is a local variable.

[1 mark]

.....

.....

14.3 State how many parameters the subroutine `applyDiscount` has.

[1 mark]

15 A school weather station records outdoor temperatures in degrees Celsius. A program is needed to analyse a set of readings entered by a student.

Write a Python program that:

- repeatedly asks the user to enter a temperature reading (a real number) until the user enters 999
- outputs Out of range if a reading is less than -30 or greater than 50, and does not use that reading
- when 999 is entered, outputs the number of valid readings, the highest valid reading and the mean of the valid readings
- outputs No valid readings instead if no valid readings were entered.

You should use meaningful variable name(s), correct syntax and indentation in your answer.

[8 marks]

16 A dice game is played by one player. The player first enters a target score, which must be between 20 and 50 inclusive – any target outside this range must be rejected and re-entered.

The player then takes turns until their score is equal to or greater than the target. In each turn:

- the computer rolls two six-sided dice by generating two random whole numbers between 1 and 6 inclusive
- if either dice shows a 1, the player scores 0 points for the turn
- otherwise, if both dice show the same number, the player scores double the total of the two dice
- otherwise, the player scores the total of the two dice
- the values of both dice and the player's score so far are output.

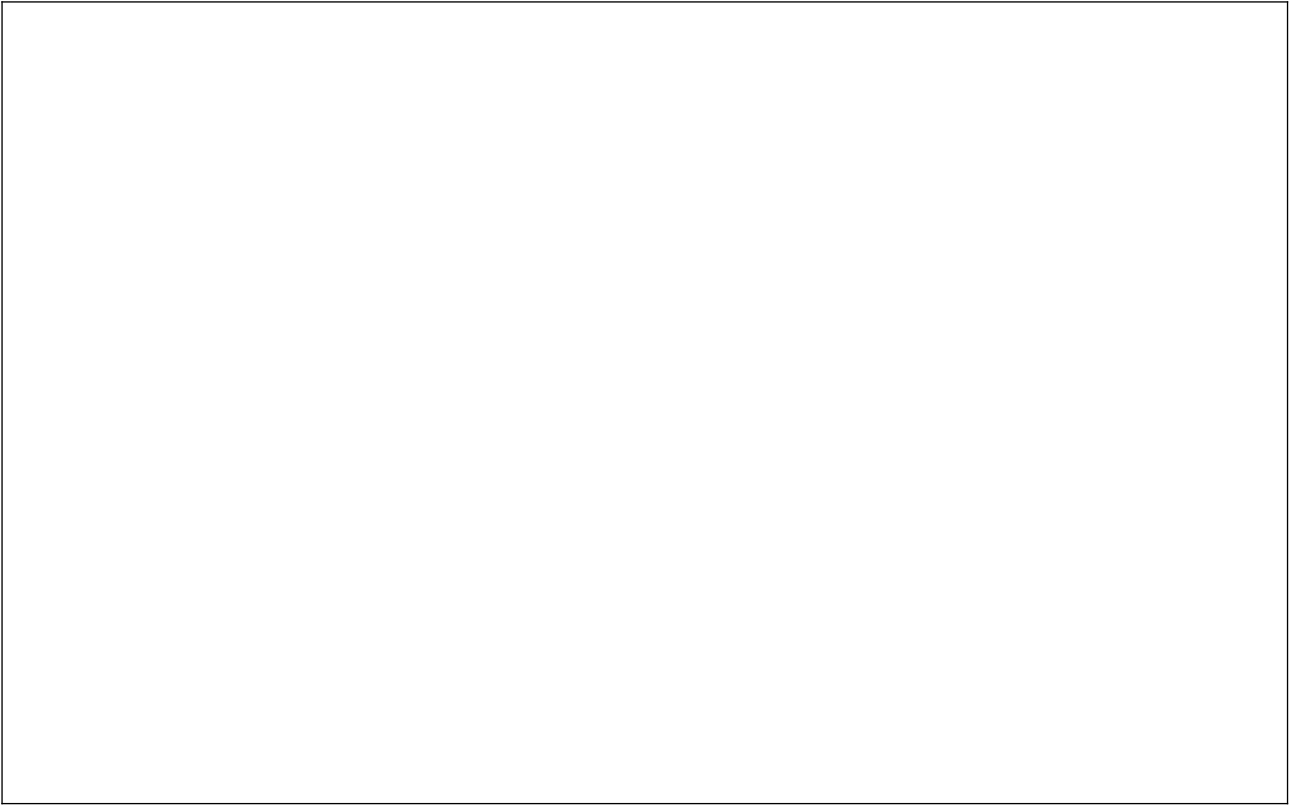
When the target has been reached, the number of turns taken is output, followed by Fast finish if 5 or fewer turns were taken, otherwise Finished.

You can generate a random number in Python using `random.randint(a, b)`, which returns a random integer between a and b inclusive. The random module has been imported for you.

Write a Python program to play the game.

You should use meaningful variable name(s), correct syntax and indentation in your answer.

[11 marks]



END OF QUESTIONS