AQA

AS
COMPUTER SCIENCE

Paper 1

Tuesday 21 May 2019 Morning Time allowed: 1 hour 45 minutes

Materials
For this paper you must have:
- a computer
- a printer
- appropriate software
- the Electronic Answer Document
- an electronic version and a hard copy of the Skeleton Program
- an electronic version of the Data Files
- an electronic version and a hard copy of the Preliminary Material.

You must **not** use a calculator.

Instructions
- Type the information required on the front of your Electronic Answer Document.
- Before the start of the examination make sure your **Centre Number, Candidate Name** and **Candidate Number** are shown clearly **in the footer** of every page (not the front cover) of your Electronic Answer Document.
- Enter your answers into the Electronic Answer Document.
- Answer **all** questions.
- Save your work at regular intervals.

Information
- The marks for questions are shown in brackets.
- The maximum mark for this paper is 75.
- No extra time is allowed for printing and collating.
- The question paper is divided into **three** sections.

Advice
You are advised to allocate time to each section as follows:
Section A – 20 minutes; **Section B** – 25 minutes; **Section C** – 60 minutes.

At the end of the examination
Tie together all your printed Electronic Answer Document pages and hand them to the Invigilator.

Warning
It may not be possible to issue a result for this paper if your details are not on every page of your Electronic Answer Document.

Section A

You are advised to spend no more than **20 minutes** on this section.

Enter your answers to **Section A** in your Electronic Answer Document. You **must save** this document at regular intervals.

Question 3 in this section asks you to write program code **starting from a new program/project/file**.

You are advised to save your program at regular intervals.

0 1 State **one** difference between local and global variables and give **two** reasons why it is good practice to use local variables.

[3 marks]

0 2 The algorithm, represented using pseudo-code in **Figure 1**, describes a method to rearrange three numbers in a data structure.

Figure 1

```
Numbers[0] ← 43
Numbers[1] ← 17
Numbers[2] ← 85
FOR x ← 1 TO 2
  MyValue ← Numbers[x]
  y ← x - 1
  WHILE (y > -1) AND (Numbers[y] < MyValue)
    Numbers[y + 1] ← Numbers[y]
    y ← y - 1
  ENDWHILE
  Numbers[y + 1] ← MyValue
ENDFOR
```

0 2 . 1 Complete **Table 1** by hand-tracing the algorithm in **Figure 1**. You may not need to use all the rows in **Table 1**.

Table 1

x	MyValue	y	y > -1 ? (True/ False)	Numbers[y]	Numbers[y] < MyValue ? (True/ False)	Numbers		
						[0]	[1]	[2]
						43	17	85

Copy the contents of all the unshaded cells in **Table 1** into your Electronic Answer Document.

[4 marks]

0 2 . 2 What type of rearrangement does this algorithm perform?

[1 mark]

Turn over for the next question

0 3 The algorithm, represented using pseudo-code, in **Figure 2** outputs a numeric result. The numeric result depends upon the value entered by the user.

Figure 2

```
OUTPUT "Enter a positive whole number: "
INPUT NumberIn
NumberOut ← 0
Count ← 0
WHILE NumberIn > 0
  Count ← Count + 1
  PartValue ← NumberIn MOD 2
  NumberIn ← NumberIn DIV 2
  FOR i ← 1 TO Count - 1
    PartValue ← PartValue * 10
  ENDFOR
  NumberOut ← NumberOut + PartValue
ENDWHILE
OUTPUT "The result is: " NumberOut
```

Table 2 lists the `MOD` and `DIV` operators for each of the available programming languages. You should refer to the row for your programming language.

Table 2

Programming language	**MOD**	**DIV**
C#	%	/
Java	%	/
Pascal	mod	div
Python	%	//
VB.Net	Mod	\

What you need to do:

Task 1
Write a program to implement the algorithm in **Figure 2**.

Task 2
Test that your program works:

- run your program, then enter the number `22`
- run your program, then enter the number `29`
- run your program, then enter the number `-1`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

0 3 . 1 Your PROGRAM SOURCE CODE for **Task 1**.

[11 marks]

0 3 . 2 SCREEN CAPTURE(S) showing the test described in **Task 2**.

[1 mark]

0 3 . 3 What is the purpose of this algorithm?

[1 mark]

Turn over for the next section

Section B

You are advised to spend no more than **25 minutes** on this section.

Enter your answers to **Section B** in your Electronic Answer Document. You **must save** this document at regular intervals.

These questions refer to the **Preliminary Material** and the **Skeleton Program**, but do **not** require any additional programming.

Refer **either** to the **Preliminary Material** issued with this question paper **or** your electronic copy.

0 4 State the name of an identifier for:

0 4 . 1 a variable that is used to store a Boolean value.

[1 mark]

0 4 . 2 a user-defined subroutine that returns a single Boolean value.

[1 mark]

0 5 The Skeleton Program uses several data structures.

State the identifier of the data structure that stores values of more than one data type.

[1 mark]

0 6 What is the specific purpose of the exception handling construct in the subroutine `SetUpBoard`?

[1 mark]

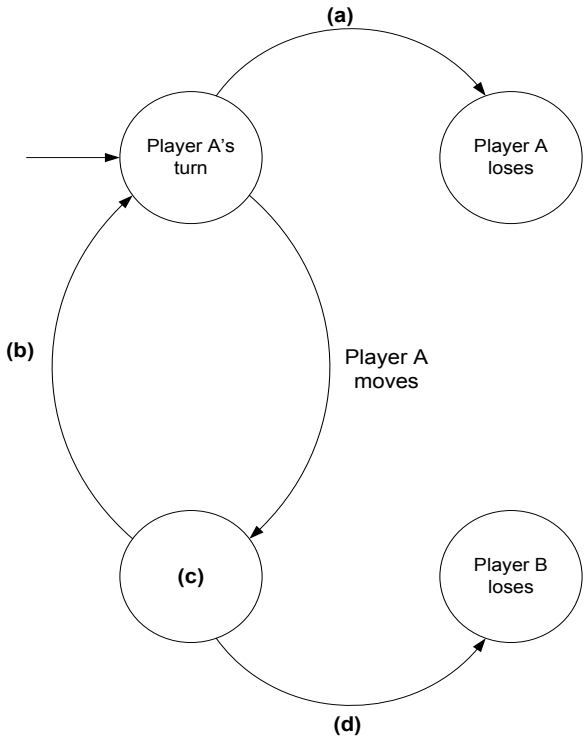
0 7 This question refers to the subroutines `Game` and `ListPossibleMoves`.

Describe what is contained in the parameter `PlayersPieces` when `ListPossibleMoves` is called for the first time in the subroutine `Game`.

[1 mark]

0 8 **Figure 3** shows an incomplete state transition diagram for the AQA Board Game. With reference to the game rules complete **Table 3**.

Figure 3



Complete **Table 3** by filling in the unshaded cells with the correct description for **Figure 3**.

Table 3

Label	Description
(a)	
(b)	
(c)	
(d)	

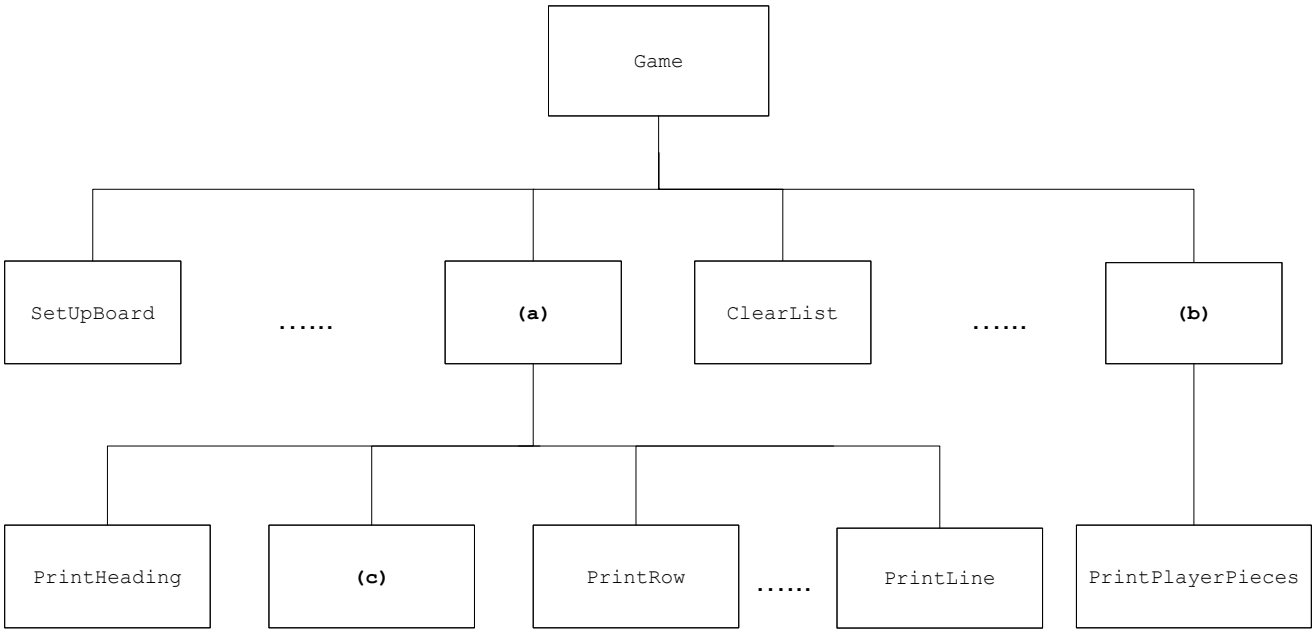
Copy the contents of all the unshaded cells in **Table 3** into your Electronic Answer Document.

[2 marks]

0 9 **Figure 4** shows an incomplete hierarchy chart for part of the **Skeleton Program**.

With reference to the **Skeleton Program** and **Figure 4**, answer questions **9.1** to **9.3**.

Figure 4



0 9 . 1 What should be written in box **(a)** in **Figure 4**?

[1 mark]

0 9 . 2 What should be written in box **(b)** in **Figure 4**?

[1 mark]

0 9 . 3 What should be written in box **(c)** in **Figure 4**?

[1 mark]

1 0 This question refers to the data structure `A` defined in the `Game` subroutine.

1 0 . 1 Explain the purpose of each of the first two values stored in row 0 in this data structure.

[2 marks]

1 0 . 2 Explain why there are 12 rows after row 0 in this data structure.

[1 mark]

1 0 . 3 Explain the purpose of the values stored in these 12 rows.

[2 marks]

1 1 This question refers to the subroutine `CreateNewBoard`.

Describe what the selection structure in this subroutine does.

[3 marks]

1 2 This question refers to the subroutine `ListPossibleMoves`.

Explain the uses of the variable `NumberOfMoves`.

[2 marks]

1 3 This question refers to the subroutine `SelectMove`.

Explain what happens in the first `WHILE` loop that terminates when a valid piece has been found.

[5 marks]

Turn over for the next section

Section C

You are advised to spend no more than **60 minutes** on this section.

Enter your answers to **Section C** in your Electronic Answer Document. You **must save** this document at regular intervals.

These questions require you to load the **Skeleton Program** and to make programming changes to it.

1 4 This question refers to the subroutine `SelectMove`. This subroutine makes three calls to the subroutine `DisplayErrorCode` to notify the user of errors. The error codes passed as parameters are to be made more informative for the user.

What you need to do:

Task 1
Modify the subroutine `DisplayErrorCode` so that meaningful error messages are displayed for each type of error explaining the circumstances which caused each error. The error code should also be displayed.

Task 2
Test that the changes you have made work by conducting the following test:

- run the program
- enter `Y`
- load `game1.txt`
- enter the string `a4`
- enter the string `a9`
- enter `3`
- enter `4`
- enter `a`
- enter `9`
- enter `3`
- enter `0`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 4 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `DisplayErrorCode`.
[3 marks]

1 4 . 2 SCREEN CAPTURE(S) showing the requested test including the list of possible moves for Player A.
[1 mark]

1 5 This question refers to the subroutine `ValidJump`. The rules of the game are to be amended. Instead of jumping over their own piece, a player can only jump over an opponent's piece.

What you need to do:

Task 1
Amend the subroutine `ValidJump` so that a jump is only possible if the middle piece belongs to the opponent.

Task 2
Test that the changes you have made work by conducting the following test:

- run the program
- enter `Y`
- load `game3.txt`
- enter the string `a5`
- enter `5`
- enter `0`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 5 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `ValidJump`.

[2 marks]

1 5 . 2 SCREEN CAPTURE(S) showing the requested test including the list of possible moves **before** the jump and your test input, and then the board display **after** the jump.

[1 mark]

Turn over for the next question

1 6 This question refers to the subroutine `PrintResult`. The rules of the game are to be amended. The game still ends when a player cannot make a move, but the winner is to be determined using a formula that calculates a score for each player. The winner is the player with the **lowest** score. The formula to calculate the score is shown in **Figure 5**.

Figure 5

Player's score = Number of that player's moves **minus** total number of player's pieces on the board **minus** number of that player's dames **multiplied by** 10

For example, if Player A made 40 moves and has 12 pieces on the board, and 3 of them are dames, then Player A's score is 40 – 12 – (3 × 10) = -2

Note that a dame is also considered to be a piece.

What you need to do:

Task 1
Create a new subroutine `CountNumberOfPieces`. This subroutine is to count the number of pieces a player has on the board.

Task 2
Amend the subroutine `PrintResult` to implement the formula given in **Figure 5** and output the score for each player and display the winner.

You must consider the possibility of a draw.

Task 3
- run the program
- enter `Y`
- load `game4.txt`

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 6 . 1 Your PROGRAM SOURCE CODE for the entire subroutine `CountNumberOfPieces` and the entire subroutine `PrintResult`.

[9 marks]

1 6 . 2 SCREEN CAPTURE(S) showing all the output from the requested test including the board display.

[1 mark]

1 7 This question will further change the rules of the game.

When a piece is promoted to a dame, the player who the new dame belongs to now chooses **one** of the opponent's pieces. This piece is removed from the board and the dame is placed in the square the removed piece was in.

1 7 . 1 State the identifier of the data structure that now needs to be passed as a parameter into the subroutine `MoveDame`.

[1 mark]

What you need to do:

Task 1
Amend the subroutine `MoveDame`.

This subroutine is to:

- ask the player the piece ID of the opponent's piece they want to remove
- check that the piece is an opponent's piece and is on the board
- remove the opponent's piece
- return the coordinates for the new dame.

Task 2
Amend the calls to `MoveDame` from within the subroutine `MovePiece`.

You will need to amend the parameter list of the subroutine heading of `MovePiece` and the call to `MovePiece` from within the subroutine `MakeMove`.

Task 3
Test that the changes you have made work by conducting the following test:

- run the program
- enter `Y`
- load `game3.txt`
- move `a2` to row 7, column 0
- take piece `b1`

Task 4

- move `b5` to row 0, column 3
- take piece `a6`

Question 17 continues on the next page

Evidence that you need to provide
Include the following evidence in your Electronic Answer Document.

1 7 . 2 Your PROGRAM SOURCE CODE for the entire subroutine `MoveDame` and the entire subroutine `MakeMove`.

[9 marks]

1 7 . 3 SCREEN CAPTURE(S) showing the requested test including the board display after piece `b1` has been taken.

[1 mark]

1 7 . 4 SCREEN CAPTURE(S) showing the requested test including the board display after piece `a6` has been taken.

[1 mark]

END OF QUESTIONS

There are no questions printed on this page

There are no questions printed on this page