General Certificate of Education Advanced Subsidiary Examination Jan 2016 (taken from June 2010)

**Computing Paper 1 (Programming) AS Jan Mock**

**For this paper you must have:**

! access to the Electronic Answer Document

! a copy of the *Preliminary Material*. You must **not** use a calculator.

**Time allowed**

! 1 Hour 30 Mins

**Instructions**

! Type your answers into the Electronic Answer Document.

! Enter the information required on the front of your Electronic Answer Document.

! Answer **all** questions.

! You will need access to:

– a computer

– a printer

– appropriate software

– the electronic version of the Skeleton Program.

! Before the start of the examination make sure your **Centre Number, Candidate Name** and **Number**

are shown clearly in the footer of the Electronic Answer Document (not the front over).

**Information**

! The marks for questions are shown in brackets.

! The maximum mark for this paper is 56.

! No extra time is allowed for printing and collating.

! The question paper is divided into four sections.

You are advised to spend time on each section as follows:

Section A - 35 minutes

Section B - 15 minutes

Section C - 40 minutes

**At the end of the examination**

! Tie together all your printed Electronic Answer Document pages and hand them to the invigilator.

**Total Score 56..**

**Section A**

You are advised to spend no more than **35 minutes** on this section.

Type your answers to **Section A** in your Electronic Answer Document.

You **must save** this document at regular intervals.

The first question in this section asks you to write program code

**starting from a new program/project/file.**

• Save your program/project/Þ le in its own folder/directory.

• You are advised to save your program at regular intervals.

**Question 1**

Create a folder/directory **Question1** for your new program.

The variable table, **Table 2**, and the Structured English algorithm, **Figure 4**, describe a simplified version of a noughts and crosses match. A match consists of a user-specified number of games. In this simplified version, the two players complete each game on paper and then enter information about the result of each game into a program that

totals the number of games won by each player. Assume that all games have a winner –

there are no drawn games.

**Table 2**

|  |  |  |
| --- | --- | --- |
| **Identifier** | **Data Type** | **Purpose** |
| NoOfGamesInMatch | Integer | Stores the number of games in the match (specified by  user) |
| NoOfGamesPlayed | Integer | Stores the number of games played so far |
| PlayerOneScore | Integer | Stores the number of games won by Player One |
| PlayerTwoScore | Integer | Stores the number of games won by Player Two |
| PlayerOneWinsGame | Char | Stores a 'Y' if Player One won the game and 'N' otherwise |

**Figure 4**

PlayerOneScore <- 0

PlayerTwoScore <- 0

OUTPUT "How many games?" INPUT NoOfGamesInMatch

FOR NoOfGamesPlayed <- 1 TO NoOfGamesInMatch Do

OUTPUT "Did Player One win the game (enter Y or N)?"

INPUT PlayerOneWinsGame

IF PlayerOneWinsGame = 'Y'

THEN PlayerOneScore <- PlayerOneScore + 1

ELSE PlayerTwoScore <- PlayerTwoScore + 1

ENDIF ENDFOR

OUTPUT PlayerOneScore

OUTPUT PlayerTwoScore

**What you need to do**

Write a program for the above algorithm.

Test the program by showing the results of a match consisting of three games where

Player One wins the first game and Player Two wins the second and third games.

Save the program in your new **Question1** folder/directory.

**Evidence that you need to provide**

*Include the following in your Electronic Answer Document.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **1** |  | Your PROGRAM SOURCE CODE. | *(9 marks)* |
| **1** | **2** |  | SCREEN CAPTURE(S) for the test described above. | *(4 marks)* |

**Question 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **1** | **3** |  | Explain what is meant by an *algorithm*. |
|  |  |  |  | One way of checking that an algorithm is correct is to complete a dry run. |
|  | **1** | **4** |  | Dry run the algorithm in **Figure 3** by completing **Table** **1.** |

*(2 marks)*

*Copy* ***all seven*** *rows of your completed* ***Table 1*** *into the table provided in the Electronic Answer Document.*

Assume that x has a value of 7.

The MOD operator calculates the remainder resulting from an integer division.

**Figure 3**

Answer 🡨 True

FOR Count = 2 TO (x – 1) DO

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | Remainder 🡨 x MOD Count |
|  |  |  | IF Remainder = 0 THEN |
|  |  |  | Answer 🡨 False |
|  |  |  | ENDIF |
|  |  | ENDFOR | |

**Table 1**

|  |  |  |
| --- | --- | --- |
| **Answer** | **Count** | **Remainder** |
| True | - | - |
|  | 2 | 1 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*(6 marks)*

**1 5**

What is the purpose of this algorithm? *(1 mark)*

**Section B**

You are advised to spend no more than **15 minutes** on this section.

Type 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 require you to load

the **Skeleton Program**, but do not require any additional programming.

R efer either to the *Preliminary Material* issued with this question paper or your

electronic copy.

**Question 3**

**1 8**  State the name of an identifier used for a global variable that has been declared in the Skeleton Program. *(1 mark)*

**1 9**  State the name of an identifier used for a local variable that has been declared in the

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | | Skeleton Program. *(1 mark)* |
|  | **2** | **0** |  | Explain a difference between a global variable and a local variable. *(2 marks)* |
|  |  |  |  | Look at the instructions in the main program block used to choose Player One’s symbol. |
|  | **2** | **1** |  | Describe the circumstances under which these instructions will stop being repeated. |
|  |  |  |  | *(2 marks)* |
|  |  |  |  |  |
|  |  |  |  | When the Skeleton Program is run it is possible that a game might stop after 9 moves  while there are still empty cells on the board – even though neither player has won. |
|  | **2** | **2** |  | Explain why this could happen. *(2 marks)* |
|  | **2** | **3** |  | State the name of an identifier for a variable that controls the number of iterations in a loop. *(1 mark)* |
|  | **2** | **4** |  | State the name of an identifier for a variable that has a constant value. *(1 mark)* |
|  |  |  |  |  |

**Turn over**

**Section C**

You are advised to spend no more than **40 minutes** on this section.

Type your answers to **Section C** in your Electronic Answer Document.

You **must save** this document at regular intervals.

T hese questions require you to load the **Skeleton Program** and make

programming changes to it.

**Question 4**

This question refers to the subroutine CheckValidMove.

This subroutine is used to check that the coordinates entered by a player are for a valid move. A valid move is defined as being an x coordinate and a y coordinate for a cell that exists and that is currently empty. At the moment the subroutine only checks that the x coordinate entered by the user is in the allowed range.

Adapt the program source code for the subroutine CheckValidMove so that it checks that the y coordinate entered by the user is in the allowed range and that the cell chosen by the user is empty.

**Evidence that you need to provide**

*Include the following in your Electronic Answer Document.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **3** | **1** |  | Your amended PROGRAM SOURCE CODE for the subroutine |  |
|  |  |  |  | CheckValidMove. | *(5 marks)* |
|  | **3** | **2** |  | SCREEN CAPTURE(S) for test runs showing that moves with coordinates |  |
|  |  |  |  | (2, -3) and (2, 7) are both rejected. | *(2 marks)* |
|  | **3** | **3** |  | SCREEN CAPTURE(S) for a test run showing that when the player selects |  |
|  |  |  |  | a non-empty cell the move is rejected. | *(1 mark)* |

**Question 5**

This question refers to the subroutine CheckXOrOHasWon.

This subroutine is used to check, after each move, if the player has won the game.

The subroutine checks for three symbols in a line on the rows and on the columns. It

should also detect three symbols in a line on the two diagonals.

Adapt the program source code for the subroutine CheckXOrOHasWon so that it

does check for **three** symbols in a line along the diagonals.

**Evidence that you need to provide**

*Include the following in your Electronic Answer Document.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **3** | **4** |  | Your amended PROGRAM SOURCE CODE for the subroutine |  |
|  |  |  |  | CheckXOrOHasWon. | *(6 marks)* |
|  | **3** | **5** |  | SCREEN CAPTURE(S) showing a game won by a player getting three in a |  |
|  |  |  |  | line along a diagonal. | *(1 mark)* |

**3 6**  SCREEN CAPTURE(S) showing a game won by a player getting three in a line along

the other diagonal. *(1 mark)*

**Question 6**

This question refers to the main program block. Part of the main program block updates the scores and displays the result using a selection structure.

Half a point should be awarded to each player if the game is drawn.

Adapt this part of the program source code to award points for a draw.

**Evidence that you need to provide**

*Include the following in your Electronic Answer Document.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | **3** | **7** |  | Your amended PROGRAM SOURCE CODE for the selection structure. | *(2 marks)* |
| **3** | **8** |  | SCREEN CAPTURE(S) showing the correct points awarded for a drawn |  |
|  |  |  |  |  | game that is the **first and only** game in a match. | *(2 marks)* |
|  |  |  |  |  |  |  |

**Skeleton** **Program**. *(2 marks)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | The Noughts and Crosses game has been adapted so that it is played using a 4x4 grid |
|  |  |  |  | on a square. It is decided to alter the program further so that it is played using a 4x4x4 |
|  |  |  |  | cube instead of a 4x4 square. |
|  | **3** | **9** |  | Describe how the data structure(s) for a cube-shaped board could be represented in the |

|  |  |
| --- | --- |
| 4 0 | Discuss possible challenges to the creation of a playable 4x4x4 game (*2 marks*) |

**END OF QUESTIONS**