General Certificate of Education Advanced Subsidiary Examination April/May 2014 (Based on 2014 Card Predict skeleton Code)

**Computing COMP1 AS 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

! 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 - 20 minutes

 Section B - 20 minutes

 Section C - 10 minutes

 Section D - 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 ..**

M/Jun10/COMP1

**COMP1**

**Section A**

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

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

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

**Question 1**







**Question 2**

 



**Question 3**



 

**Turn over for the next section**

**Section B**

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

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

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

 The 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 4**

 Create a folder/directory **Question4** for your new program.

The variable table, the table below, and the Structured English algorithm below, describe a simplified version of a game of guessing Heads or Tails by flipping a coin.

|  |  |  |
| --- | --- | --- |
| **Identifier** | **Data type** | **Purpose** |
| CurrentGuess | Character | Stores the Guess of either H or T entered by the user |
| CurrentFlip | Character | Stores either H or T To represent Heads or Tails as the result of a simulated coin toss. |
| NumberOfCorrectGuesses | Integer | Stores an integer that holds the number of correct guesses the user has made. Should be initialized as zero |
| FlipSequence | String | Stores a string of H or T characters that represent a series of results of Tossing a coin. It should be hard coded to hold “XHTHTT” (the X is there so that you can treat this as a string array starting at element1) |
| NumberOfFlips | Integer | A constant that holds the maximum number of flips to be tossed during the game. It should be set to 5 |

LOOP

 OUTPUT "Please Guess Heads (H) or Tails (T)"

 INPUT CurrentGuess

 OUTPUT "Coin Fliping...."

 CurrentFlip = FlipSequence(NumberOfCorrectGuesses + 1)

 OUTPUT "The coin landed "

 OUTPUT CurrentFlip

 IF CurrentGuess = CurrentFlip

 THEN

 OUTPUT "well done. Let’s flip again."

 NumberOfCorrectGuesses = NumberOfCorrectGuesses + 1

 IF NumberOfCorrectGuesses = NumberOfFlips

 OUTPUT "Congratulations You completed the sequence. Game Over"

 ENDIF

 ELSE OUTPUT "sorry you guessed incorrectly. GAME OVER")

 ENDIF

UNTIL CurrentFlip <> CurrentGuess Or NumberOfCorrectGuesses = NumberOfFlips

OUTPUT "you guessed " NumberOfCorrectGuesses " flips correctly"

**What you need to do**

Write a program for the above algorithm.

Test the program by playing the game twice.
(i)Once where the user correctly guesses the first flip and then incorrectly guesses the second flip.
(ii)Secondly where the user correctly guesses all 5 flips.
Save the program in your new folder/directory.

**Evidence that you need to provide**

|  |  |
| --- | --- |
| **1** | **9** |

(a)     Your PROGRAM SOURCE CODE.

**(9)**

|  |  |
| --- | --- |
| **2** | **0** |

SCREEN CAPTURE(S)The first test described above, where the user correctly guesses the first flip and then incorrectly guesses the second flip **(2)**

|  |  |
| --- | --- |
| **2** | **1** |

SCREEN CAPTURE(S) where the user correctly guesses all 5 flips **(1)**

**Turn over for the next section**

**Section C**

You are advised to spend no more than **10 minutes** on this section.

 Type your answers to **Section C** 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 5**

 Give the name of an identifier that has a fixed value role. *(1 mark)*

 Give an example of a variable that has the role of a most recent holder *(1 mark)*

 State the name of a user-defined function that has exactly 2 parameters *(1 mark)*

 What are the 2 fields of the structure TRecentScore? *(1 mark)*

 The first 6 lines of the file deck.txt are as follows:



 Look at the subroutine LoadDeck.

The deck has been loaded without shuffling and the game has not yet been played.

 What value is held in Deck(2).Rank?

 *(1 mark)*

 To answer this question, you may find it helpful to look at the subroutines PlayGame and GetCard.

The game has now been played and the player has scored 4. What value is now held in Deck(50).Suit?

 *(1 mark)*

The following decision table shows the logic which determines whether a player has guessed correctly in the PlayGame subroutine.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Conditions | (a) | True | True | False | False |
| Choice = “y” | True | False | True | False |
| Choice = “n” | False | True | False | True |
| Action | Output correct guess messageReset card | X |  | (b) | (c) |
| Game over |  | X | (d) | (e) |

 What should be written in the cell labelled (a)? *(1 mark)*

 Which cell(s) out of (b), (c), (d) and (e) should also have an ‘X’ in them? *(1 mark)*

 There is a variable called Count in the GetCard subroutine.

There is also a variable called Count in the UpdateRecentScores subroutine.

Explain why these two different variables can have the same identifier. *(2 marks)*

**Turn over**

**Section D**

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

 Type your answers to **Section D** 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 6**

**This Question refers to the Function** IsNextCardHigher

Currently Only the RANK of a card is used to assess if one card is higher that another.
ALTER THE CODE so that the suit of the card is used to determine which of two cards of the same rank is higher- suits are ranked 1 to 4 – Clubs =1, Diamonds=2, Hearts=3, Spades=4

THUS: . “Four Of Spades” is Higher than “Four of Clubs”

To test the change to the program you will conduct 2 tests

1. Using OPTION 2 in the game to play without shuffle
Guess using the following sequence: **y,n,y,y,n**
2. Using OPTION 2 in the game to play without shuffle
Guess using the following sequence: **y,y**

**Evidence that you need to provide**

|  |  |
| --- | --- |
| **3** | **4** |

Your amended PROGRAM SOURCE CODE for function IsNextCardHigher (3 Marks)

|  |  |
| --- | --- |
| **3** | **5** |

SCREEN CAPTURE of Test 1 above

( you may need to stretch the console window in order to see the whole sequence) (2 Marks)

|  |  |
| --- | --- |
| **3** | **6** |

SCREEN CAPTURE of Test 1 above (1 mark)

**Question 7**

**This question refers to** UpdateRecentScores

At the moment the three most recent scores are store. When a fourth game is played its score is stored at the expense of the first score recorded.

Change the code so that a the scores are stored in order, the highest being in position 1 of the array and therefore being displayed at the top of any list that is output.

Test the program by playing the game four times and then displaying the output of recentscores Play the game a fifth time (ensure that you score more than the lowest score currently shown. And display the recentscore table again

|  |  |
| --- | --- |
| **3** | **7** |

Your amended PROGRAM SOURCE CODE for Procedure UpdateRecentScores

|  |  |
| --- | --- |
| **3** | **8** |

SCREEN CAPTURE of the Test above

**END OF QUESTION**