1.1 Programming part 2

Name: ______________________

Class: ______________________

Date: ______________________

Time: **660 minutes**

Marks: **516 marks**

Comments:

Q1.

State the name of an identifier for:

(a) a user-defined data type.

___

___

(1)

(b) a built-in function that converts a string into a different data type.

___

___

(1)

(c) an array variable.

___

___

(1)

A variable can have one of a number of different roles. Some of the different possible roles a variable can have are:

- fixed value
- stepper
- gatherer
- transformation
- follower
- temporary
- most recent holder
- most wanted holder.

EXAM PAPERS PRACTICE

For each of the following variables state which of the possible roles best describes the role of the variable:

(d) `SwapSpace` in the `ShuffleDeck` subroutine.

___

___

(1)

(e) `Choice` in the `PlayGame` subroutine.

___

___

(1)

(f) `NoOfCardsTurnedOver` in the `PlayGame` subroutine.

(1)

An extra subroutine that could have been added to the **Skeleton Program** is `HasPlayerXGotARecentScore`. This subroutine would have looked at the contents of the `RecentScores` array and output a message saying if someone with a particular name has (or has not) got one of the recent scores.

The code below shows a first attempt (written in pseudo-code) to develop an algorithm that the `HasPlayerXGotARecentScore` subroutine could be based on.

```
OUTPUT "Enter a name"
INPUT PlayerX

Found ← False

Position ← 1
WHILE Found = False DO
  IF RecentScores[Position].Name = PlayerX

    THEN Found ← True

    ELSE Position ← Position + 1
  ENDIF
ENDWHILE
IF Found = True
  THEN OUTPUT "Yes, they do have a recent score"
  ELSE OUTPUT "No, they do not have a recent score"
ENDIF
```

The algorithm shown above is then implemented in a programming language. There is an error in the algorithm which means that when the program is run it sometimes works correctly and sometimes it does not.

(g) Under what circumstances will a program based on the algorithm shown not work as intended?

EXAM PAPERS PRACTICE

(1)

(h) How should the algorithm above be changed so that this problem is corrected?

You may answer this question by either describing the change(s) needed or by giving a new version of the relevant part(s) of the pseudo-code for the algorithm above.

(2)

(i) State the name of the algorithm that is being used as a basis for the development of the HasPlayerXGotARecentScore subroutine.

____

____

(1)

(Total 10 marks)

Q2.

(a) This question refers to the subroutine GetPlayerName.

Add a validation check to the subroutine GetPlayerName so that it repeatedly attempts to get the name from the user until a name with at least one character in it is entered (the name cannot be left blank).

Each time an invalid value is entered the message "You must enter a name" should be displayed.

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

- run the **Skeleton Program**
- select option 2 from the menu
- play a game
- when the prompt "Please enter your name: " is displayed press the Enter key without entering a name
- then enter Emily as a name.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine GetPlayerName.

(4)

(ii) SCREEN CAPTURE(S) showing the requested test.

EXAM PAPERS PRACTICE

(2)

(b) This part refers to the subroutine IsNextCardHigher.

The game is to be altered so that if two cards have the same rank then the suit of the cards determines which of the two cards is the higher. Spades is the highest suit, then hearts, then diamonds, then clubs. For example:

- if the last card was the 7 of Diamonds and the next card is the 7 of Hearts then the subroutine IsNextCardHigher should return a value of True
- if the last card was the 7 of Diamonds and the next card is the 7 of Clubs then the subroutine IsNextCardHigher should return a value of False.

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

- run the **Skeleton Program**
- select option 2 from the menu
- when asked if you think the next card will be higher enter y, then n, then y.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine

`IsNextCardHigher`.

(4)

(ii) SCREEN CAPTURE(S) showing the requested test.

(2)

(c) This part will extend the functionality of the game.

The game is to be altered so that the player can play a joker. When asked if they think the next card will be higher the player can enter a `j` to play a joker instead of guessing `y` or `n`. When the player uses a joker it doesn't matter what the next card is as the player is considered to have predicted correctly whether the next card is higher or not.

The player can play a joker a maximum of two times in a game.

Task 1
Adapt the `GetChoiceFromUser` subroutine so that an appropriate message is displayed that informs the user how to play a joker.

Evidence that you need to provide
(i) Your amended PROGRAM SOURCE CODE for the subroutine `GetChoiceFromUser`.

(1)

Task 2
Adapt the `PlayGame` subroutine so that the player can play a joker in the way described.

Test your program works by conducting the following test:

- run the **Skeleton Program**
- select option 2 from the menu
- when asked if you think the next card will be higher enter `j`, then `j`, then `j`.

EXAM PAPERS PRACTICE

Evidence that you need to provide
(ii) Your amended PROGRAM SOURCE CODE for the subroutine `PlayGame`.

(7)

(iii) SCREEN CAPTURE(S) showing the requested test.

(3)

(d) This part will further extend the functionality of the game.

The game is to be altered so that the player is told the probability that the next card will be higher than the last card. Each time the player is asked to make a prediction, they should first be shown the probability that the next card will be higher than the last card.

The probability of the next card being higher than the last card can be calculated by performing the division:

$$\frac{\text{Number of cards not yet turned over that are higher than the last card turned over}}{\text{Number of cards not yet turned over}}$$

Additional marks will be awarded in this question for writing code that demonstrates good practice by ensuring subroutines are self-contained and make use of interfaces.

Task 1

Create a new subroutine, `CalculateProbability`, which works out the probability of the next card being higher than the last card. It should return this calculated value to the calling routine. You may choose whether to make the new subroutine a function or a procedure.

Evidence that you need to provide

(i) Your PROGRAM SOURCE CODE for the subroutine `CalculateProbability`.

(11)

Task 2

Adapt the `PlayGame` subroutine so that, before the second call to the `GetCard` subroutine, the message "`The probability of the next card being higher is X`" is displayed, where `X` is the value returned by the `CalculateProbability` subroutine.

Test your program works by conducting the following test:

- run the **Skeleton Program**
- select option 2 from the menu
- the probability of the next card being higher is displayed to the user
- when asked if you think the next card will be higher enter `y`
- the probability of the next card being higher is displayed to the user.

Evidence that you need to provide

(ii) Your amended PROGRAM SOURCE CODE for the subroutine `PlayGame`.

(3)

(iii) SCREEN CAPTURE(S) showing the requested test.

EXAM PAPERS PRACTICE

(2)

(Total 39 marks)

Q3.

Create a folder / directory for your new program.

The algorithm, represented using pseudo-code in **Figure 1**, and the variable table, **Table 1**, describe a simple two player game. Player One chooses a whole number between 1 and 10 (inclusive) and then Player Two tries to guess the number chosen by Player One. Player Two gets up to five attempts to guess the number. Player Two wins the game if they correctly guess the number, otherwise Player One wins the game.

Note that in **Figure 1**, the symbol < > means "is not equal to".

Figure 1

```
OUTPUT "Player One enter your chosen number: "
INPUT NumberToGuess
WHILE NumberToGuess < 1 OR NumberToGuess > 10 DO
      OUTPUT "Not a valid choice, please enter another number: "
      INPUT NumberToGuess
```

```
ENDWHILE
Guess ← 0
NumberOfGuesses ← 0
WHILE Guess <> NumberToGuess AND NumberOfGuesses < 5 DO
      OUTPUT "Player Two have a guess: "
      INPUT Guess
      NumberOfGuesses ← NumberOfGuesses + 1
ENDWHILE
IF Guess = NumberToGuess
   THEN OUTPUT "Player Two wins"
   ELSE OUTPUT "Player One wins"
```

Table 1

Identifier	Data type	Purpose
`NumberToGuess`	`Integer`	Stores the number entered by Player One
`NumberOfGuesses`	`Integer`	Stores the number of guesses that Player Two has made so far
`Guess`	`Integer`	Stores the most recent guess made by Player Two

What you need to do

Write a program for the above algorithm.

Test the program by conducting the tests **Test 1** and **Test 2**.

Save the program in your new folder / directory.

Test 1
Test that your program works correctly by conducting the following test:

EXAM PAPERS PRACTICE

- Player One enters the number 0
- Player One enters the number 11
- Player One enters the number 5
- Player Two enters a guess of 5

Test 2
Test that your program works correctly by conducting the following test:

- Player One enters the number 6
- Player Two enters guesses of 1, 3, 5, 7, 10

Evidence that you need to provide

(a) Your PROGRAM SOURCE CODE.

(13)

(b) SCREEN CAPTURE(S) showing the result of **Test 1**.

(4)

(c) SCREEN CAPTURE(S) showing the result of **Test 2**.

(3)

Part of the algorithm from **Figure 1** is shown in **Figure 2.**
Note that in **Figure 2**, the symbol <> means "is not equal to".

Figure 2

```
WHILE Guess < > NumberToGuess AND NumberOfGuesses < 5 DO
    OUTPUT "Player Two have a guess: "
    INPUT Guess

    NumberOfGuesses ← NumberOfGuesses + 1
ENDWHILE
```

(d) Explain why a `WHILE` repetition structure was chosen instead of a `FOR` repetition structure for the part of the algorithm shown in **Figure 2**.

____

____

(1)

(Total 21 marks)

Q4.

Throughout this question, you must be careful to copy and paste or type accurately the names of identifiers from the **Skeleton Program**.

State the name of an identifier for:

(a) a variable used to store whole numbers.

____

____

(1)

(b) a user-defined subroutine that has exactly **three** parameters.

____

____

(1)

(c) a built-in function with exactly **one** parameter that returns an integer value.

____

____

(1)

(d) Give an example of an assignment statement from the **Skeleton Program** where avariable is assigned an empty string.

Look at the *option j* in the main program block.

(1)

(e) Explain why `AmountToShift` needs to be assigned the value of `– GetKeyForCaesarCipher`.

(2)

(f) Look at the `ApplyShiftToASCIICodeForCharacter` subroutine.

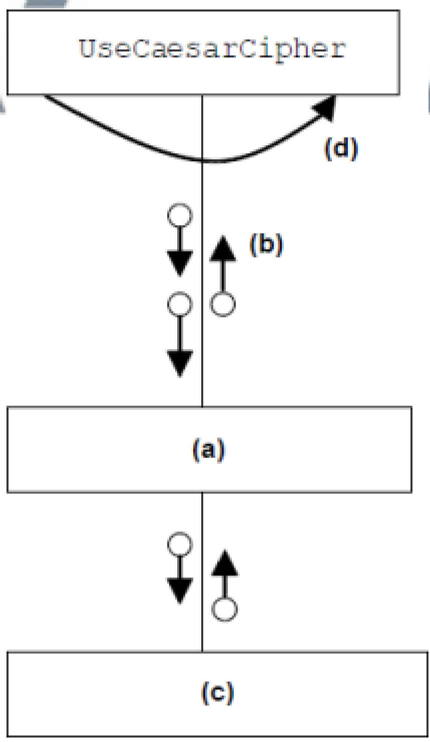
Explain, using an example, why `Mod 26` is used when calculating `NewASCIICode`.

(2)

EXAM PAPERS PRACTICE

Figure 1 shows an **incomplete** structure chart for part of the **Skeleton Program**.

Figure 1



(g) What should be written in box **(a)** in **Figure 1**?

____

____

(1)

(h) How should the arrow **(b)** in **Figure 1** be labelled?

____

____

(1)

(i) What should be written in box **(c)** in **Figure 1**?

____

____

(1)

(j) How should the curved arrow **(d)** in **Figure 1** be labelled?

____

____

(1)

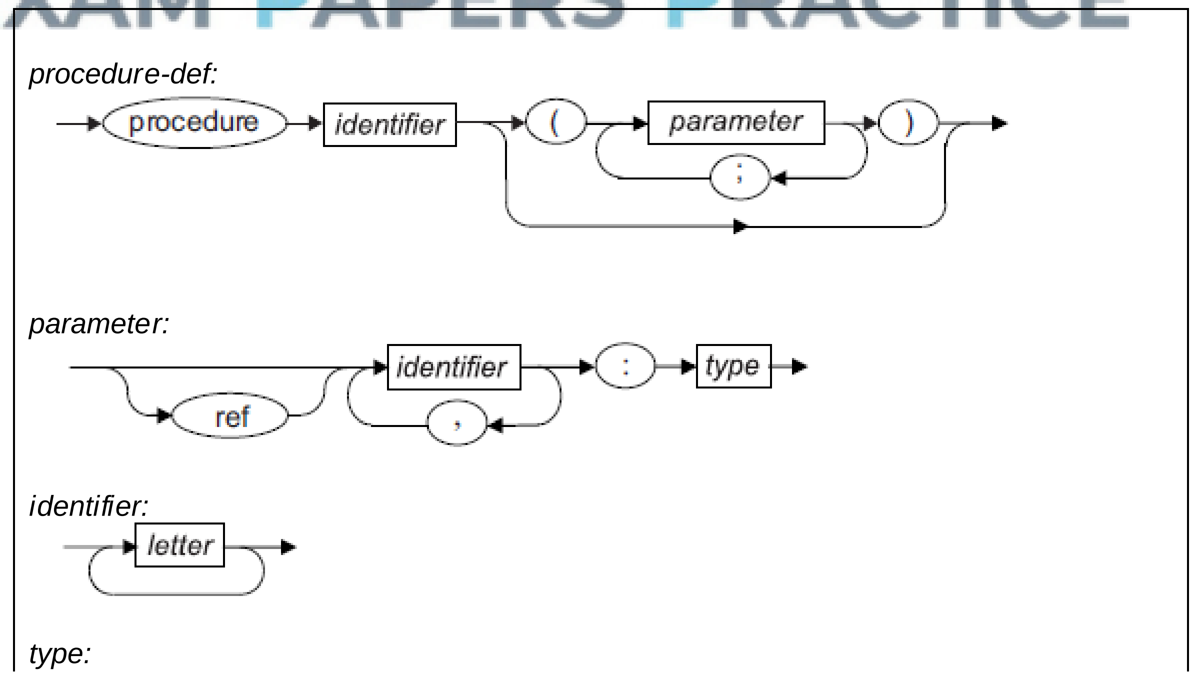
(Total 12 marks)

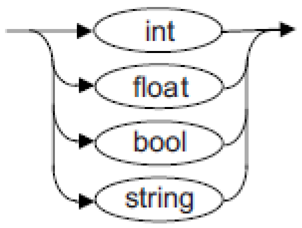
Q5.

In a particular programming language, the correct syntax for four different constructs is defined by the syntax diagrams in **Figure 1.**

Figure 1

EXAM PAPERS PRACTICE





A *letter* is any alphabetic character from "a" to "z" or "A" to "Z".

In this language an example of a valid *identifier* is `loopCount` and an example of a valid *type* is `int`.

(a) For each row in the table below, write **Yes** or **No** in the empty column to identify whether or not the **Example** is a valid example of the listed **Construct**.

Construct	Example	Valid? (Yes / No)
identifier	`Player2name`	
parameter	`x,y:bool`	
procedure-def	`procedure square(s:real)`	
procedure-def	`procedure rect(w:int,h:int)`	

(4)

(b) A student has written Backus-Naur Form (BNF) production rules that are supposed to define the same constructs as the syntax diagrams in **Figure 1**. Their BNF rules are shown in **Figure 2**.

Figure 2

```
<procedure-def> ::= procedure <identifier> ( <paramlist> )
<paramlist>     ::= <parameter> | <parameter> ; <paramlist>
<parameter>     ::= <identlist> : <type> |
                    ref <identlist> : <type>
<identlist>     ::= <identifier> | <identifier> , <identlist>
<identifier>    ::= <letter> | <letter> <identifier>
<type>          ::= int | float | bool | char | string
```

EXAM PAPERS PRACTICE

A <letter> is any alphabetic character from "a" to "z" or "A" to "Z".

(i) The BNF production rules in **Figure 2** contain two errors. These errors mean that they do not represent the same statement types as the syntax diagrams in **Figure 1.**

Describe the **two** errors.

Error 1:____

____

Error 2:____

____

(2)

(ii) The production rule for a <paramlist> is recursive.

Explain why recursion has been used in this production rule.

____

____

____

____

(1)

(Total 7 marks)

Q6.

The table below is a partially complete representation of the rules for adding together two bit values. The first two columns represent the two bit values to add. The first row has been completed and represents the binary addition rule `0 + 0 = 0`. Carry occurs when the answer cannot be stored in 1 bit.

		Answer	Carry
0	0	0	0
0	1		
1	0		
1	1		

Complete the table above to show the **Answer** and **Carry** values for the given binary addition rules.

(Total 3 marks)

EXAM PAPERS PRACTICE

Q7.

The ASCII system uses 7 bits to represent a character. The ASCII code in denary for the numeric character '0' is 48; other numeric characters follow on from this in sequence.

(a) Using 7 bits, express the ASCII code for the character '2' in binary.

Characters are transmitted using an 8-bit code that includes a single parity bit in the most significant bit. A parity bit is added for error checking during data transmission.

____

(1)

(b) Using odd parity, what 8-bit code is sent for the numeric character '0'?

____

(2)

Hamming code is an alternative to the use of a single parity bit.

(c) State **one** advantage of using Hamming code instead of a single parity bit.

____

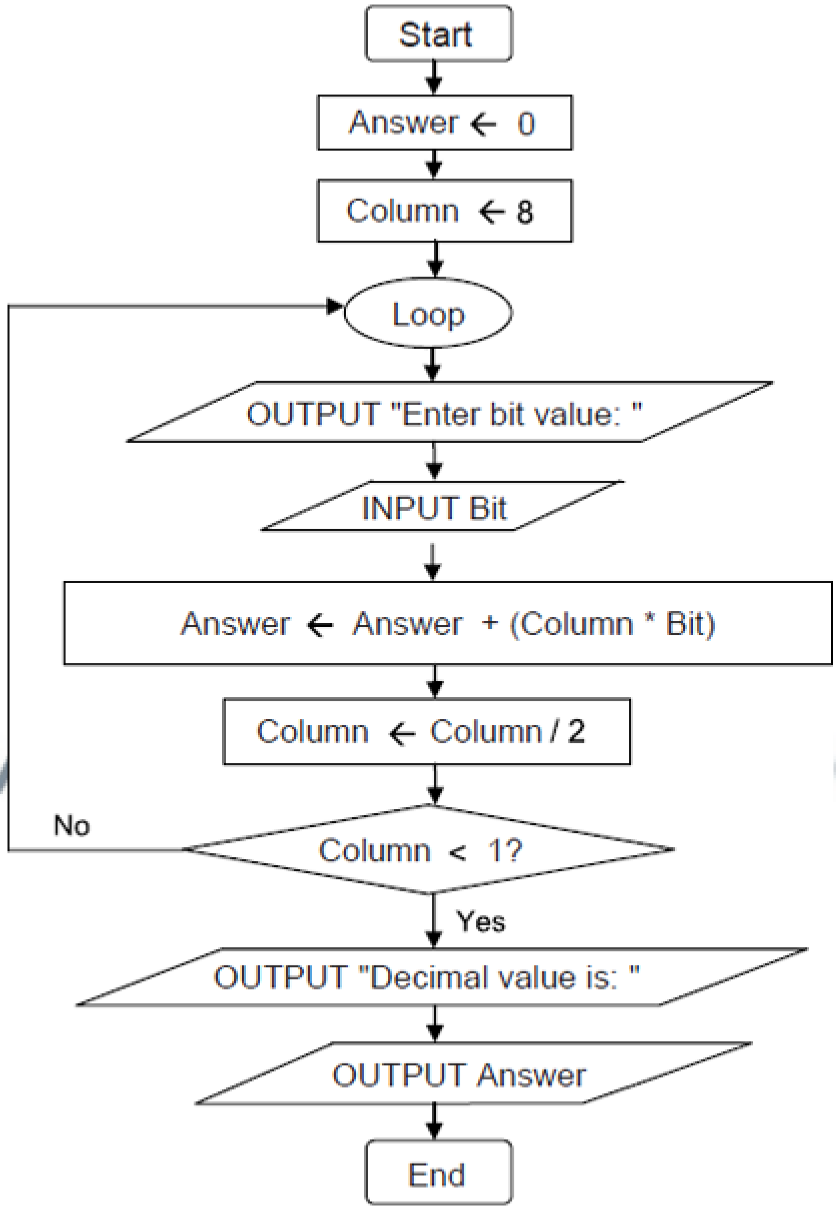
____

(1)
(Total 4 marks)

Q8.

Create a folder/directory for your new program.

The algorithm, represented as a flowchart below, and the variable table, describe the converting of a 4-bit binary value into denary.



Identifier	**Data type**	**Purpose**
`Column`	`Integer`	Stores the place value (column heading)
`Answer`	`Integer`	Stores the denary value equivalent to the bit pattern entered by the user

`Bit`	`Integer`	Stores a `0` or `1` entered by the user

What you need to do

Write a program for the above algorithm.

Test the program by showing the result of entering the values `1, 1, 0, 1 (in that order).`

Save the program in your new folder/directory.

Evidence that you need to provide

(a) Your PROGRAM SOURCE CODE.

(11)

(b) SCREEN CAPTURE(S) for the test described above.

(3)

(c) What is the largest denary number that could be output by the algorithm represented by the flowchart in the diagram above?

____

(1)

(d) The algorithm represented by the flowchart above can convert sixteen different bit patterns into denary.

If the symbol $\boxed{\text{Column} \leftarrow 8}$ is changed to $\boxed{\text{Column} \leftarrow 16}$ how many **more** bit patterns could be converted into denary?

____

EXAM PAPERS PRACTICE

(1)

(e) When developing a new system the stages of the systems development life cycle could be followed.

At which stage of the systems development life cycle would the flowchart above have been created?

____

(1)

(f) At which stage of the systems development life cycle would the algorithm represented by the flowchart above be automated using a programming language?

____

(1)

(Total 18 marks)

Q9.

State the name of an identifier for:

(a) a user-defined subroutine that has only one parameter.

____

(1)

(b) user-defined subroutine whose only action is to produce output to the screen.

____

(1)

(c) a variable that has a stepper role.

____

(1)

(d) an array variable.

____

(1)

(e) Look at the repetition structure in the `SetPositionOfItem` subroutine.

Describe the circumstances under which this structure in the **Skeleton Program** will stop repeating.

____

____

____

____

____

(3)

EXAM PAPERS PRACTICE

(f) Look at the `SetUpGame` subroutine.

Why has a `For` loop been chosen for the repetition structure?

____

____

(1)

(g) The `For` loop repeats `NoOfTrap` times.

Why has a named constant ben used instead of the numeric value 2?

____

____

(1)

(h) When a game is saved it is stored as a binary file. A text file could have been used instead.

Describe a difference between the way that data are stored in a binary file and the way that data are stored in a text file.

___

___

___

___

(2)

(i) The subroutines in the **Skeleton Program** avoid the use of global variables – they use local variables and parameter passing instead.

State **two** reasons why subroutines should, ideally, not use global variables.

___

___

___

___

(2)

(j) Below is a pseudo-code representation of the part of the `PlayGame` subroutine that is used to check if the player has triggered one of the traps in the cavern.

```
MonsterAwake ← CheckIfSameCell(PlayerPosition,
                    TrapPositions[1])
If Not MonsterAwake
   Then MonsterAwake ← CheckIfSameCell(PlayerPosition,
                        TrapPositions[2])
EndIf
```

Why is it necessary that the check for the triggering of the second trap is inside the selection structure?

___

___

___

___

(2)

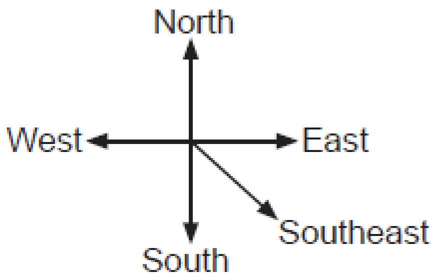
(Total 15 marks)

Q10.

(a) This question refers to the subroutines `DisplayMoveOptions`, `CheckValidMove` and `MakeMove`.

The player can currently move in four directions – north, south, west and east. The player is to be allowed to move diagonally.

Adapt the program source code for the subroutines `DisplayMoveOptions`, `CheckValidMove` and `MakeMove` so that there is a fifth direction – southeast (as shown in the diagram below) – that can be selected by entering a "`D`".



Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine `DisplayMoveOptions`. **(1)**

(ii) Your amended PROGRAM SOURCE CODE for the subroutine `MakeMove`. **(3)**

(iii) Your amended PROGRAM SOURCE CODE for the subroutine `CheckValidMove`. **(1)**

(iv) SCREEN CAPTURE(S) for a test run showing the correct working of the new move option being selected and the player moving to the southeast. **(2)**

(b) This question refers to the subroutines `CheckValidMove` and `PlayGame`.

The **Skeleton Program** currently does not make all the checks needed to ensure that the move entered by a player is an allowed move. It should **not** be possible to make a move that takes a player outside of the 7x5 cavern grid.

The **Skeleton Program** needs to be adapted so that it prevents a player from moving north if they are at the northernmost end of the cavern.

The subroutine `CheckValidMove` needs to be adapted so that it returns a value of `False` if a player attempts to move north when they are at the northernmost end of the cavern.

The subroutine `PlayGame` needs to be adapted so that it displays an error message to the user if an illegal move is entered. The message should state "`That is not a valid move, please try again`".

Evidence that you need to provide.

(i) Your amended PROGRAM SOURCE CODE for the subroutine `PlayGame`. **(3)**

(ii) Your amended PROGRAM SOURCE CODE for the subroutine `CheckValidMove`. **(4)**

(iii) SCREEN CAPTURE(S) for a test run showing a player trying to move north when they are at the northernmost end of the cavern. **(1)**

(c) This question refers to the `PlayGame` subroutine and will extend the functionality of

EXAM PAPERS PRACTICE

the game.

The number of moves made by a player in a game of MONSTER! will be tracked. A variable called `NoOfMoves` will be used to store the number of moves made by a player.

The final number of moves made will be displayed to the user at the end of the game.
At the end of the game, either the player will have found the flask or the player will have been eaten by the monster.

If they have found the flask then a message should be displayed saying "`The number of moves you took to find the flask was` *X*" – where *X* is the value of `NoOfMoves`.

If they were eaten then a message should be displayed saying "`The number of moves you survived in the cavern for was` *X*" – where *X* is the value of `NoOfMoves`.

Task 1
Create a new variable, of an appropriate data type, called `NoOfMoves`. **At the start of a game** an initial value of `0` should be assigned to the `NoOfMoves` variable.

Task 2
The value of `NoOfMoves` needs to be incremented **after** a player has completed a move in the cavern.

Task 3
Adapt the relevant subroutine(s) so that the correct messages are displayed at the end of a game of MONSTER!

Task 4 – Test 1
Test that the changes you have made work by conducting the following test:

- Play the training game
- Move south
- Move south
- Move east

Task 5 – Test 2
Test that the changes you have made work by conducting the following test:

- Play the training game
- Move south
- Move west

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the `PlayGame` subroutine and (if relevant) the PROGRAM SOURCE CODE for any other subroutine(s) you have amended.

(5)

EXAM PAPERS PRACTICE

(ii) SCREEN CAPTURE(S) showing the result of **Test 1**.

(1)

(iii) SCREEN CAPTURE(S) showing the result of **Test 2**.

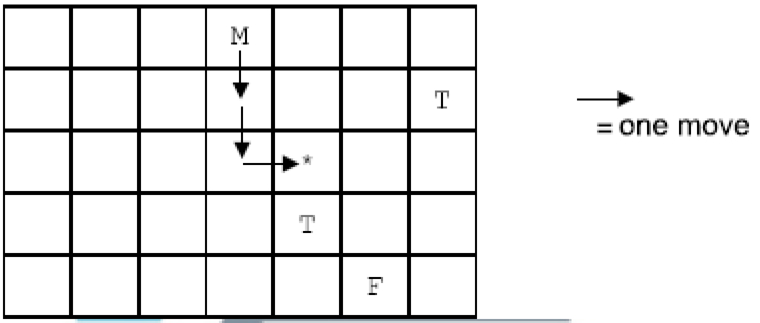
(1)

(d) This question will extend the functionality of the game.

The noise made by the sleeping monster does not help the player work out in which direction the monster is – however, it gets louder as the player moves nearer to the monster and gets quieter as the player moves further away from the monster.

The game is to be adapted so that after the move options have been displayed (but before the user enters their move) a message is displayed stating the distance between the monster and the player.

The *distance* between the monster and the player is measured by the number of cells the monster would have to move into in order to get to the cell currently occupied by the player. For example, at the start of the training game (diagram below in the **Preliminary Material**, reproduced below) the distance would be 3 as the monster would have to move into 3 cells in order to get to the player's cell.



Additional marks will be awarded in this question for writing code that demonstrates good practice by ensuring subroutines are self-contained and make use of interfaces.

EXAM PAPERS PRACTICE

Task 1
Create a new subroutine, `CalculateDistance`, which works out the distance between the cell currently occupied by the monster and the cell currently occupied by the player. It should then return this calculated value to the calling routine.

Evidence that you need to provide

(i) Your PROGRAM SOURCE CODE for the new subroutine `CalculateDistance`.

(7)

Task 2
Adapt the `PlayGame` subroutine so that it displays (after the move options have been shown) the message "`Distance between monster and player: X`" – where *X* is the distance between the monster and the player.

Test that your program works by loading the training game and showing that:

- the correct distance is displayed before the player's first move
- the correct distance is displayed after the player's first move, one cell to the north

- the correct distance is displayed after the player's third move, both second and third moves are westwards.

Evidence that you need to provide

(ii) Your amended PROGRAM SOURCE CODE for the `PlayGame` subroutine.

(3)

(iii) SCREEN CAPTURE(S) showing the distance message and the cavern at the start of the training game, before the player's first move.

(1)

(iv) SCREEN CAPTURE(S) showing the distance message and the cavern after the player has moved one cell to the north.

(1)

(v) SCREEN CAPTURE(S) showing the distance message and the cavern after the player has then moved two cells to the west.

(1)

(Total 35 marks)

Q11.

A graph can be drawn to represent a maze. In such a graph, each graph vertex represents one of the following:

- the entrance to or exit from the maze
- a place where more than one path can be taken
- a dead end.

Edges connect the vertices according to the paths in the maze.

Diagram 1 shows a maze and **Diagram 2** shows one possible representation of this maze.

EXAM PAPERS PRACTICE

Position 1 in **Diagram 1** corresponds to vertex 1 in **Diagram 2** and is the entrance to the maze. Position 7 in **Diagram 1** is the exit to the maze and corresponds to vertex 7.

Dead ends have been represented by the symbol in **Diagram 2**.

Diagram 3 shows a simplified undirected graph of this maze with dead ends omitted.

Diagram 1

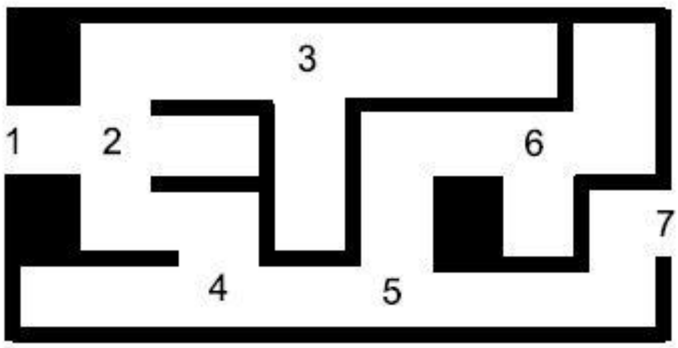
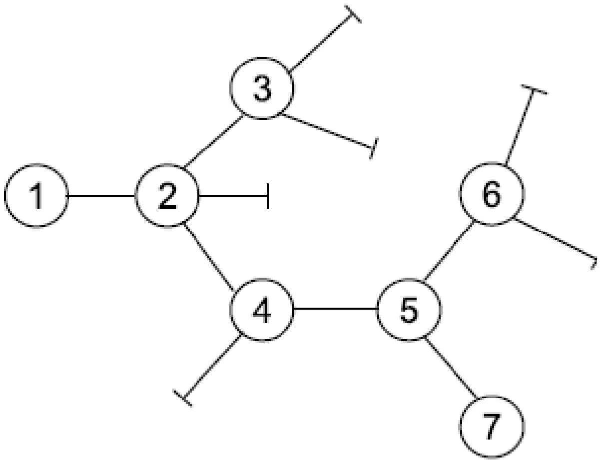
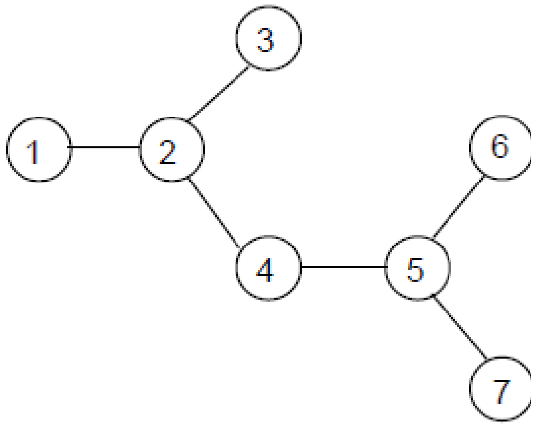


Diagram 2



Representation of maze
including dead ends

Diagram 3



Graph representing maze
with dead ends omitted

(a) The graph in **Diagram 3** is a tree.

State **one** property of the graph in **Diagram 3** that makes it a tree.

____

____

(1)

(b) The graphs of some mazes are not trees.

Describe a feature of a maze that would result in its graph **not** being a tree.

____

____

(1)

EXAM PAPERS PRACTICE

(c) Complete the table below to show how the graph in **Diagram 3** would be stored using an adjacency matrix.

(2)

(d) (i) What is a *recursive routine?*

____

____

(1)

(ii) To enable the use of recursion a programming language must provide a stack.

Explain what this stack will be used for and why a stack is appropriate.

EXAM PAPERS PRACTICE

____

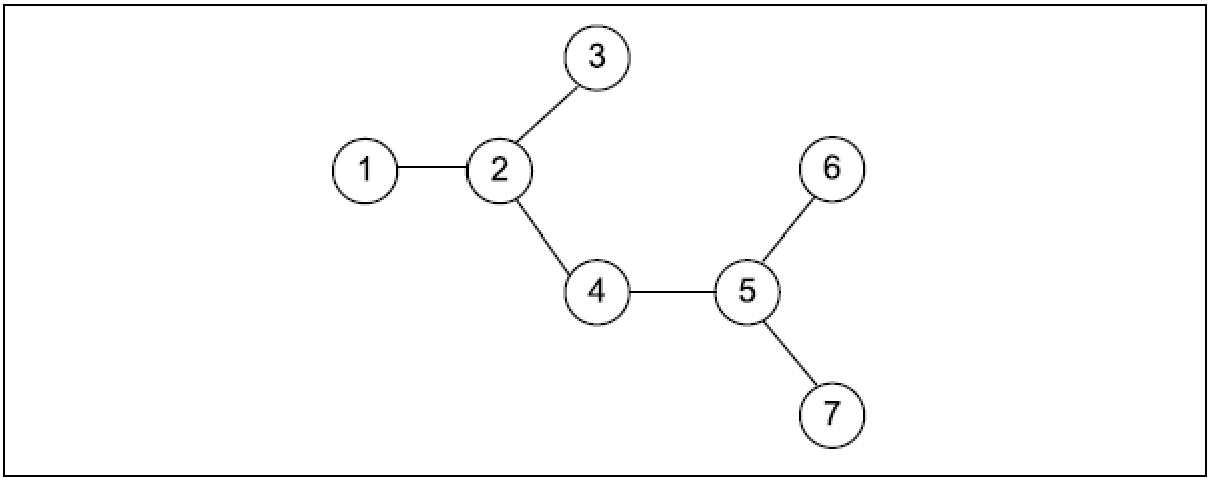
____

____

____

(2)

Diagram 3 is repeated here so that you can answer Question (e) without having to turn pages.



(e) A recursive routine can be used to perform a depth-first search of the graph that represents the maze to test if there is a route from the entrance (vertex 1) to the exit (vertex 7).

The recursive routine in the diagram below is to be used to explore the graph in **Diagram 3**. It has two parameters, `V` (the current vertex) and `EndV` (the exit vertex).

```
Procedure DFS(V, EndV)
    Discovered[V] ← True
    If V = EndV Then Found ← True
    For each vertex U which is connected to V Do
        If Discovered [U] = False Then DFS(U, EndV)
    EndFor
    CompletelyExplored[V] ← True
EndProcedure
```

Complete the trace table below to show how the `Discovered` and `CompletelyExplored` flag arrays and the variable `Found` are updated by the algorithm when it is called using `DFS(1,7)`.

EXAM PAPERS PRACTICE

The details of each call and the values of the variables `V`, `U` and `EndV` have already been entered into the table for you. The letter `F` has been used as an abbreviation for `False`. You should use `T` as an abbreviation for `True`.

				Discovered							CompletelyExplored							
Call	V	U	EndV	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	Found
	-	-		F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
DFS(1,7)	1	2	7															
DFS(2,7)	2	1	7															
		3	7															
DFS(3,7)	3	2	7															
DFS(2,7)	2	4	7															
DFS(4,7)	4	2	7															
		5	7															
DFS(5,7)	5	4	7															
		6	7															
DFS(6,7)	6	5	7															
DFS(5,7)	5	7	7															
DFS(7,7)	7	5	7															
DFS(5,7)	5	-	7															
DFS(4,7)	4	-	7															
DFS(2,7)	2	-	7															
DFS(1,7)	1	-	7															

(5)
(Total 12 marks)

EXAM PAPERS PRACTICE

Q12.

Create a folder/directory for your new program.

The variable table, the table given below, and the Structured English algorithm, the diagram below, describe a linear search algorithm that could be used with a simplified version of the Dice Cricket game to find out if a particular player's name appears in the high score table.

In this simplified version only the names of the players getting a top score are stored. Their scores are **not** stored.

Identifier	Data Type	Purpose
`Names`	`Array[1..4] of String`	Stores the names of the players who have one of the top scores
`PlayerName`	`String`	Stores the name of the player being looked for
`Max`	`Integer`	Stores the size of the array

`Current`	`Integer`	Indicates which element of the array `Names` is currently being examined
`Found`	`Boolean`	Stores `True` if the player's name has been found in the array, `False` otherwise

```
Names[1] ← 'Ben'
Names[2] ← 'Thor'
Names[3] ← 'Zoe'
Names[4] ← 'Kate'
Max ← 4
Current ← 1
Found ← False
OUTPUT 'What player are you looking for?'
INPUT PlayerName
WHILE (Found = False) AND (Current <= Max)
  IF Names[Current] = PlayerName
    THEN Found ← True
    ELSE Current ← Current + 1
  ENDIF
ENDWHILE
IF Found = True
  THEN OUTPUT 'Yes, they have a top score'
  ELSE OUTPUT 'No, they do not have a top score'
ENDIF
```

What you need to do

- Write a program for the above algorithm.
- Test the program by searching for a player named 'Thor'.
- Test the program by searching for a player named 'Imran'.
- Save the program in your new folder/directory.

EXAM PAPERS PRACTICE

Evidence that you need to provide.

(a) Your PROGRAM SOURCE CODE.

(11)

(b) SCREEN CAPTURE(S) for the test searching for 'Thor'.

(2)

(c) SCREEN CAPTURE(S) for the test searching for 'Imran'.

(2)

(Total 15 marks)

Q13.

A constant is a value that does not change throughout a program. Instead of referring to the value itself throughout a program, a named constant can be used.

(a) Give an example of a constant declaration from the **Skeleton Program**.

____

(1)

(b) State **one** advantage of using named constants for constant values.

(1)

(c) State the name of an identifier for a variable that has a fixed value role.

(1)

(d) State the name of an identifier for a variable that has a most wanted holder role.

(1)

The decision table shown below represents the logic of the selection structure in the `GetMenuChoice` subroutine. '✓' has been used to indicate the action that results from particular values for the conditions. The decision table is only partially complete; some incomplete parts have been labelled **(a)**, **(b)**, **(c)** and **(d)**

Conditions	`OptionChosen <1`	`True`	`False`	`False`	`False`
	`OptionChosen > 4`	`False`	`True`	`True`	**(d)**
	`OptionChosen <> 9`	**(c)**	`False`	`True`	`True`
Action	Output error message	✓	**(a)**	**(b)**	

EXAM PAPERS PRACTICE

(e) Which of the two cells labelled **(a)** and **(b)** should have a '✓' in it?

(1)

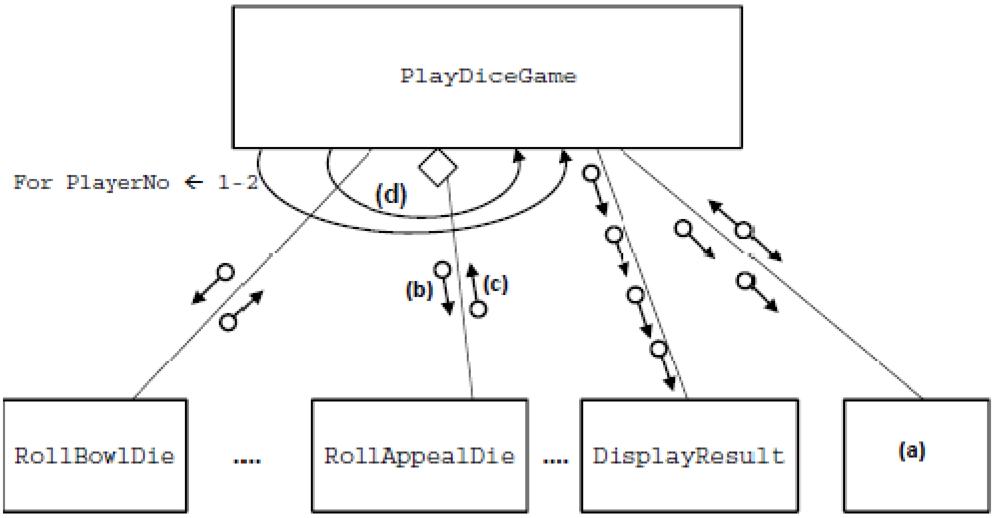
(f) What should be the contents of the cell labelled **(c)**?

(1)

(g) What should be the contents of the cell labelled **(d)**?

(1)

The diagram below shows an incomplete structure chart for part of the **Skeleton Program.**



With reference to the **Skeleton Program** and the diagram above, answer questions h to k.

(h) What should be written in box **(a)** in figure above?

____ **(1)**

(i) How should the arrow **(b)** in the diagram above be labelled?

____ **(1)**

(j) How should the arrow **(c)** in the diagram above be labelled?

____ **(1)**

(k) How should the curved arrow **(d)** in the diagram above be labelled?

____ **(1)**

(l) There is a variable called `Count` in the `LoadTopScores` subroutine.
There is also a variable called `Count` in the `UpdateTopScores` subroutine.

Explain why these two different variables can have the same identifier.

____

____

____

____ **(2)**

(m) Look at the repetition structure in the `UpdateTopScores` subroutine, used to find the lowest of the current top scores.

EXAM PAPERS PRACTICE

When `UpdateTopScores` is called, how many times will this section of code repeat?

____

(1)

(n) Describe what the selection structure inside the repetition structure does.

____

____

____

____

____

____

(4)

(Total 18 marks)

Q14.

(a) This question refers to the subroutines `RollAppealDie` and `DisplayAppealDieResult`.

There are four options on the Appeal Die – `"NOT OUT"`, `"CAUGHT"`, `"LBW"` and `"BOWLED"`.

Adapt the program source code for the subroutines `RollAppealDie` and `DisplayAppealDieResult` so that there is a fifth option – "`RUN OUT`" – on the Appeal Die.

If a player is run out then their turn finishes, they are out. A suitable message must be shown.

EXAM PAPERS PRACTICE

This option should be available for **both** the real dice and virtual dice versions of the game.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine `RollAppealDie`.

(2)

(ii) Your amended PROGRAM SOURCE CODE for the subroutine `DisplayAppealDieResult`.

(2)

(iii) SCREEN CAPTURE(S) for a test run showing the correct working of the `"RUN OUT"` option when real dice are being used.

(2)

(b) This question refers to the subroutine `DisplayResult`.

This subroutine compares the two players' scores and displays a message saying who has won.

Adapt the program source code for the subroutine `DisplayResult` so that it also checks to see if a game is drawn and displays an appropriate message when this happens.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine `DisplayResult`.

(3)

(ii) SCREEN CAPTURE(S) for a test run showing a drawn game where both players scored 0.

(2)

(c) This question refers to the subroutine `RollBowlDie`.

If the user chooses to play the game with real dice then they are prompted to enter a number between 1 and 6 to indicate what the result of rolling the Bowl Die was.

Add a validation check to the subroutine `RollBowlDie` so that it repeatedly gets the Bowl Die result from the user until a number between 1 and 6 is entered.

Each time an invalid value is entered the message "`Please enter a value between 1 and 6 only`" should be displayed.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine `RollBowlDie`.

(4)

(ii) SCREEN CAPTURE(S) showing the results of testing the subroutine with values of 0, 2 and 7 for the Bowl Die.

EXAM PAPERS PRACTICE

(3)

(d) *You may wish to make a copy of the data file **HiScores.txt** before attempting this question in case the contents of the file are changed in an unintended way.*

This question will add extra functionality to the **Skeleton Program.**

The **Skeleton Program** allows two players to have a game of Dice Cricket. It can load previous top scores from the file **HiScores.txt** and every time a game is played the scores of the players are compared to the top scores. The top scores are then updated if necessary.

The **Skeleton Program** is going to be extended so that the top scores can be saved to the file **HiScores.txt**.

Additional marks will be awarded in this question for writing code which demonstrates good practice and which will be easy to maintain in the future.

Task 1
Change the `DisplayMenu` subroutine so that it displays the new menu option "`5. Save top scores`".

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the subroutine `DisplayMenu`.

(1)

Task 2
Adapt the `GetMenuChoice` subroutine so that a value of 5 is accepted.

Evidence that you need to provide

(ii) Your amended PROGRAM SOURCE CODE for the subroutine `GetMenuChoice`.

(1)

Task 3
Create a new subroutine `SaveTopScores`.

The new subroutine must:

- open the file **HiScores.txt**
- store each record in the `TopScores` array as a line in the file; with the `Name` and `Score` fields separated by a comma
- close the file **HiScores.txt**

Evidence that you need to provide

(iii) Your PROGRAM SOURCE CODE for the new subroutine `SaveTopScores`.

(10)

Task 4
Adapt the main program block so that the selection of option 5 from the menu is accepted as a valid choice and so that the subroutine `SaveTopScores` is called when the user selects option 5 from the menu.

EXAM PAPERS PRACTICE

Test that the changes you have made work:

- run the **Skeleton Program**
- enter the player names Janet and Lily
- load the contents of the file **HiScores.txt**
- play a real dice game where player one gets a score of 4 and player two gets a score of 0
- select the save option you have added to the menu.

Evidence that you need to provide

(iv) Your adapted PROGRAM SOURCE CODE for the main program block.

(4)

(v) SCREEN CAPTURE(S) for a test run showing that:

- Option 3 (load scores) was selected
- Option 2 (real dice game) was selected
- Player one (Janet) got a score of 4
- Player two (Lily) got a score of 0
- Option 5 (save scores) was selected.

(2)

(vi) Copy and paste the contents of the file **HiScores.txt** after the test run.

(1)

(e) This question refers to rolling the Bowl Die.

The game of Dice Cricket, as represented by the **Skeleton Program**, is to be made more similar to the real game of cricket. In real cricket, a batter is likely to get the lower scores (0 and 1) more frequently than the higher scores (4 and 6).

The **Skeleton Program** is to be modified to represent this more realistic batting behaviour.

(i) Describe changes that could be made to the **Skeleton Program** to achieve this. You are **not** expected to actually make the changes.

____

____

____

____

____

____

____

____

(2)
(Total 39 marks)

Q15.

A computer simulation is to be used to imitate the flow of students through a school canteen. The simulation will be based on a model developed by the school's canteen manager and a Computing student.

(a) In the context of simulation, explain what a model is.

____

____

(1)

EXAM PAPERS PRACTICE

(b) Students must queue at a particular serving point in the canteen if they wish to purchase hot food.

The Computing student intends to represent the queue of students waiting to be served as a dynamic data structure using a linked list.

(i) Explain what pointers the student will need to create and what they will be used for.

____

____

____

____

(2)

(ii) Teachers are able to bypass the students in the queue by walking past them. However, a teacher may not always go directly to the very front of the queue as it may contain teachers already. In which case, the teacher joins the queue at the point just behind the other teachers.

What type of queue would the Computing student use to represent this situation?

____

(1)

(c) The Computing student decides that she will need to use the random number generator in the programming language that she is using to develop the simulation.

Give **one** example of something that she might need to use random numbers for when producing this simulation.

EXAM PAPERS PRACTICE

____

____

(1)

(Total 5 marks)

Q16.

State **three** features of well-written program code that help to make it understandable without the need to include lots of comments.

____

____

____

____

____

____

(Total 3 marks)

Q17.

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

____

(1)

(ii) State the name of an identifier used for a local variable that has been declared in the Skeleton Program.

____

(1)

(iii) Explain a difference between a global variable and a local variable.

____

____

____

____

(2)

(iv) Look at the instructions in the main program block used to choose Player One□s symbol.

Describe the circumstances under which these instructions will stop being repeated.

____

EXAM PAPERS PRACTICE

____

____

____

(2)

(v) 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.

Explain why this could happen.

____

____

____

____

(2)

(vi) State the name of an identifier for a variable that has a stepper role.

____

(1)

(vii) State the name of an identifier for a variable that has a fixed-value role.

____

(1)

(viii) State the name of an identifier for a user-defined subroutine that has exactly **three** parameters.

____

(1)

(ix) Study the code for the subroutine `GetWhoStarts`.

Give an example of an assignment statement in this subroutine.

____

(1)

(x) Describe what the selection structure in this subroutine does.

____

____

____

____

____

____

(3)

EXAM PAPERS PRACTICE

(b) This question refers to the subroutine `CheckValidMove`.

As part of the testing of a subroutine, boundary data should be used.

(i) Explain what is meant by a *boundary value.*

____

____

(1)

(ii) List the **three** essential boundary values that should be used to test the upper boundary of the x coordinate entered by a user.

____

____

____

(3)

(iii) Include SCREEN CAPTURE(S) showing the input and output when you run the **Skeleton Program** using **one** of the values given in your answer to **part (b)(i).**

(1)
(Total 20 marks)

Q18.

(a) 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

(i) Your amended PROGRAM SOURCE CODE for the subroutine `CheckValidMove`.

(5)

(ii) SCREEN CAPTURE(S) for test runs showing that moves with coordinates (2, –3) and (2, 7) are both rejected.

(2)

(ii) SCREEN CAPTURE(S) for a test run showing that when the player selects a non-empty cell the move is rejected.

(1)

(b) 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

(i) Your amended PROGRAM SOURCE CODE for the subroutine `CheckXOrOHasWon`.

(6)

(ii) SCREEN CAPTURE(S) showing a game won by a player getting three in a line along a diagonal.

(1)

(iii) SCREEN CAPTURE(S) showing a game won by a player getting three in a line along the other diagonal.

(1)

(c) This question refers to the main program block.

Part of the main program block updates the scores and displays the result using a

EXAM PAPERS PRACTICE

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

(i) Your amended PROGRAM SOURCE CODE for the selection structure.

(2)

(ii) SCREEN CAPTURE(S) showing the correct points awarded for a drawn game that is the **first and only** game in a match.

(2)

(d) This question expands the functionality of the **Skeleton Program.** The **Skeleton Program** needs to be changed so that the game is played on a **4x4** grid instead of a 3x3 grid. The aim of the game is still to get **three** consecutive symbols in a line.

Follow the steps below to change the **Skeleton Program** to work with a 4x4 grid.

- Change the `Board` array data type.

Evidence that you need to provide

(i) Your amended PROGRAM SOURCE CODE for the necessary change to the `Board` array data type or an explanation of why no change is required.

(1)

- Change the condition for the selection structure in the main program that checks if the maximum number of allowed moves has been reached.

Evidence that you need to provide

(ii) Your amended PROGRAM SOURCE CODE for the necessary change to the condition for the selection structure in the main program.

EXAM PAPERS PRACTICE

(1)

- Change the subroutine `ClearBoard` so that it clears the 4x4 grid.

Evidence that you need to provide
(iii) Your amended PROGRAM SOURCE CODE for the subroutine `ClearBoard`.

(1)

- Change the subroutine `DisplayBoard` so that it displays the 4x4 grid.

Evidence that you need to provide
(iv) Your amended PROGRAM SOURCE CODE for the subroutine `DisplayBoard`.

(2)

(v) SCREEN CAPTURE(S) showing an empty 4x4 grid is displayed at the beginning of the game.

(2)

- Change the subroutine `CheckValidMove` so that it checks that the move entered by a user is valid on the 4x4 grid.

Evidence that you need to provide

(vi) Your amended PROGRAM SOURCE CODE for the subroutine `CheckValidMove`.

(1)

- Change the subroutine `CheckXOrOHasWon` so that it checks for three of the same symbol in consecutive positions on a horizontal line on any row on the 4x4 grid. You are **not** expected to check for the extra winning positions along the diagonals or in the columns.

Evidence that you need to provide

(vii) Your amended PROGRAM SOURCE CODE for the subroutine `CheckXOrOHasWon`.

(4)

(viii) SCREEN CAPTURE(S) showing a 4x4 grid game won with three symbols in a horizontal line - where one of the symbols in the winning line is in cell (2,4).

(2)

(ix) 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.

Describe how the data structure(s) for a cube-shaped board could be represented in the **Skeleton Program.**

____

____

____

____

(2)

(Total 36 marks)

EXAM PAPERS PRACTICE

Q19.

Create a folder/directory for your new program.

The variable table, **Table 1**, and the Structured English algorithm describe a simplified version of the **Guess the Word / Phrase Game**.

Table 1

Identifier	Data Type	Purpose
`NewWord`	`String`	Stores the setter's word to be guessed
`UserWordGuess`	`String`	Stores a word that is the user's guess

```
OUTPUT "The new word?"
INPUT NewWord
OUTPUT "Your guess?"
```

```
INPUT UserWordGuess
IF UserWordGuess IS EQUAL TO NewWord
    THEN OUTPUT "CORRECT"
    ELSE OUTPUT "INCORRECT"
ENDIF
```

What you need to do

Write a program for the above algorithm in the programming language of your choice.

Test the program as follows.

Test 1: Input of the new word EAGLE followed by a correct guess.

Test 2: Input of the new word BEAR followed by an incorrect guess.

Save the program in your new folder / directory.

Evidence that you need to provide

(a) SCREEN CAPTURES for the following tests:

(i) Test 1

(3)

(ii) Test 2

(3)

(b) Your PROGRAM SOURCE CODE.

(7)

(Total 13 marks)

Q20.

“Guess the Word / Phrase Game”

EXAM PAPERS PRACTICE

The **Skeleton Program** in this Preliminary Material is for a game based on the game of “Hangman” in which the *user* is given a certain number of letter guesses to guess a chosen word or phrase.

This exercise will not require the display of a gallows!

The game starts with the input by the *setter* of a word or phrase of **at least 10 characters** chosen from a set of characters consisting of the uppercase letters of the alphabet and the <Space> character. You may assume that the *setter* never sets a word or phrase with more than 20 characters.

The output device, i.e. the screen, then displays a row of asterisks (‘*’) corresponding to every letter in the word or phrase and a space for every space.

The game will not use words / phrases containing other characters, e.g. hyphens, apostrophes or digits (0, 1, 2, …. 9).

A second person, the *user*, must then enter a letter that they think could be present in the *setter’s* word or phrase or if they think that they recognise the word or phrase, they enter this word or phrase.

In each of these cases, what the *user* has done is **make a guess**.

If the letter guess is correct, the row of asterisks displays again with the guessed letter replacing one or more asterisks in the corresponding positions that this letter appears in the word / phrase.

If the word / phrase guess is correct, the game is over.
A message displays which states, “You have guessed correctly.”

If either guess is incorrect, the row of asterisks displays again with no change.

Restrictions

Two people are required to play this game, a *setter* and a *user*.

The *setter* inputs the word / phrase to be guessed. The *user’s* role is to guess the word / phrase either directly by submitting the word / phrase or indirectly by guessing its letters. The *user* must not have sight of the word / phrase before playing the game.

The **Skeleton Program** does not store or display all letters that the *user* enters (history of letters entered), only those that are correct guesses.

The **Skeleton Program** in its present form does not allow the user to attempt a guess of the complete word / phrase.

The game allows the *user* to make an unlimited number of letter guesses.

The letter case of the *user’s* guess must match the word / phrase’s letter case which is upper case.

Outline Design

The game begins when a new word or phrase is set. **For the purpose of the design a phrase is considered to consist of one or more words.**

The Structured English description of an algorithm for playing the game is as follows:

```
INPUT  word / phrase
check number of characters in word / phrase
IF  acceptable
 THEN
  DO
   OUTPUT  word / phrase with letters not yet guessed masked by asterisks
   INPUT  next letter guess
   IF  letter is present
     THEN  update values of variables
   ENDIF
  LOOP UNTIL  all letters guessed
 ELSE OUTPUT  "Not enough letters"
ENDIF
```

Variables

The main variables used are as follows:

Identifier	Data Type	Purpose
NewPhrase	String	The word / phrase entered by the

EXAM PAPERS PRACTICE

		setter
`GuessStatusArray`	`Array[1..20] of Char`	Stores: • an &ltAsterisk>in each position not yet guessed • the letters in positions that have been guessed
`IndividualLettersArray`[1]	`Array[1..20] of Char`	Stores: • the individual characters from `NewPhrase` • any &ltSpace>character(s) in `NewPhrase` are stored as a &ltSpace&gt

Note: The programming language used to code the game will determine the letter case for each identifier, and so may not match exactly the identifiers shown in the table above.

Guessing a Letter

The data types used by the various programming language **Skeleton Programs** differ slightly.

- All languages store the *setter's* word / phrase in a variable `NewPhrase`
- Languages such as Pascal, C, PHP and C# permit access to individual characters of the string as follows:

  E.g.
  `NewPhrase := "DERBY COUNTY";`
  `NewPhrase[1]` gives `'D'`, `NewPhrase[4]` gives `'B'`.

- Other languages such as Visual Basic.Net are unable to access the individual letters of `NewPhrase` directly.
  An array `IndividualLettersArray` is used to store the letters to permit access to individual letters as follows:

  EXAM PAPERS PRACTICE

  `NewPhrase = "DERBY COUNTY"`
  Access to the first and fourth letters is made possible as follows:
  `IndividualLettersArray[1]` gives `'D'`.
  `IndividualLettersArray[4]` gives `'B'`.

Figure 1 shows the contents of the `NewPhrase / IndividualLettersArray` for the phrase "COMPUTING EXAM".
`NewPhrase` is used for the languages which support access to individual characters.

All solutions store the letter guesses in array `GuessStatusArray`.
Figure 2 shows the contents of the array `GuessStatusArray` before the user has made any guesses.

Figure 3 shows the contents of the array `GuessStatusArray` after the user has guessed the letters `'E', 'A', 'W', 'P', 'C', 'U'` in that order.
Note: the unsuccessful guess `'W'` is not stored.

Figure 1 **Figure 2** **Figure 3**

	NewPhrase / IndividualLettersArray		GuessStatusArray		GuessStatusArray
1	C	1	*	1	C
2	O	2	*	2	*
3	M	3	*	3	*
4	P	4	*	4	P
5	U	5	*	5	U
6	T	6	*	6	*
7	I	7	*	7	*
8	N	8	*	8	*
9	G	9	*	9	*
10	<Space>	10	<Space>	10	<Space>
11	E	11	*	11	E
12	X	12	*	12	*
13	A	13	*	13	A
14	M	14	*	14	*
:	:	:	:	:	:
:	:	:	:	:	:
:	:	:	:	:	:
20		20		20	

Possible Additional Requirement

The **Skeleton Program** does not display every letter that has been entered; it only displays correctly guessed letters.

EXAM PAPERS PRACTICE

Consider how the letters entered during the game could be stored. You need only consider the two suggestions shown below.

Suggestion 1
Use a one-dimensional array `LettersGuessedArray`[2], **Figure 4**, with each array cell corresponding to a letter of the alphabet.

Figure 4

Index	1	2	3	4	5	6	7	8	9		25	26
LettersGuessedArray[2]												

For example:
`LettersGuessedArray[4]` stores an indicator that the *user* entered the letter `'D'`.

Suggestion 2
Use a one-dimensional array `LettersGuessedArray`, **Figure 5**, as follows:
Each letter entered is stored in this array at the next available cell.

For example:
`LettersGuessedArray[1]` stores the entered letter `'E'`,

`LettersGuessedArray[2]` stores the entered letter `'A'` because the *user* entered the letter 'E' first followed by the letter 'A'.

Figure 5

Index	1	2	3	4	5	6	7	8	9	…	25	26
LettersGuessedArray[2]	E	A										

An entered letter is never stored more than once.

[1] The `IndividualLettersArray` is only needed when the programming language does not support direct access to the individual letters of `NewPhrase`.

[2] Your chosen programming language may use arrays with a lower bound value of 0. If so, then assume that `LettersGuessedArray[0]` is not used.

Q21.

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

This question is about the structure and content of the **Skeleton Program**.

(a) Give **three** reasons why this program has been structured with procedures / functions.

____

____

____

____

EXAM PAPERS PRACTICE

____

____

(3)

(b) The following questions are all about the identifiers used in this program.

State the name of an identifier used for:

(i) a local variable;

____

____

(1)

(ii) a global variable;

____

____

(1)

(iii) a pre-defined function with a single parameter;

____

____

(1)

(iv) an array variable;

____

____

(1)

(v) a variable that is used to control the iteration of a loop;

____

____

(1)

(vi) a user-defined (i.e. programmer-defined) procedure / function that only produces output to the screen.

____

____

(1)

(c) (i) State the name of a user-defined procedure / function that has one or more parameters.

____

____

(1)

EXAM PAPERS PRACTICE

(ii) Name the parameter(s).

____

____

(1)

(d) The design and implementation of the **Skeleton Program** includes one validation check on the word or phrase that is input by the user.

Describe this validation check.

____

____

____

____

(3)

(e) Study the code for the function `GetNewPhrase`.

(i) What is the condition that controls the execution of the loop?

(1)

(ii) What will be the outcome if the setter continually keys in a word / phrase which fails the validation test?

(1)

(f) Procedure `SetUpGuessStatusArray` uses a `For` loop.

(i) Why is a loop needed?

(1)

(ii) Why is a `For` loop chosen?

(1)

(iii) What determines the number of iterations for a given input word / phrase?

(1)

This question requires **no coding**. The **Skeleton Program** does not store every letter guess made by the user.

The **Preliminary Material** contains two designs, labelled **Suggestion 1** and **Suggestion 2,** for storing **every** letter guess.

Study **Suggestion 1**.

(g) The user makes four guesses, `'B','E','F','J'` in that order.

State the array positions where contents have changed. What do these cells now contain?

EXAM PAPERS PRACTICE

(5)

(h) Will the stored data change if the user then enters `'F'` again, by mistake?

(1)

Study **Suggestion 2**.

(i) The user makes four guesses, `'C'`, `'G'`, `'B'`, `'H'` in that order.

State the array positions where contents have changed. What do these cells now contain?

EXAM PAPERS PRACTICE

(2)

(j) The user enters `'B'` again, by mistake.

(i) State whether or not the `LettersGuessedArray` changes.

(1)

(ii) Explain your answer to part (d)(i).

(1)
(Total 29 marks)

Q22.

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

The menu currently provides the user with three choices (1, 2 and 5).

What you need to do

Make the following amendments to the **Skeleton Program**.

- Add another choice to the menu:
  `"3. USER - Make a complete word / phrase guess"`
- Add a new procedure / function `InputUsersCompletePhraseGuess`
  Code this procedure / function as a stub, which only displays the message:
  `"Procedure InputUsersCompletePhraseGuess has been called"`
- Add the code to call this procedure when menu choice 3 is selected.

Test that the program displays the correct message when menu choice 3 is selected.

Evidence that you need to provide

(a) Your amended PROGRAM SOURCE CODE for procedure / function `DisplayMenu`.

(2)

(b) Your PROGRAM SOURCE CODE for procedure / function `InputUsersCompletePhraseGuess`.

(3)

(c) The PROGRAM SOURCE CODE STATEMENT(S) that you have written to call procedure / function `InputUsersCompletePhraseGuess`.

(2)

EXAM PAPERS PRACTICE

(d) A SCREEN CAPTURE of the test showing that the procedure / function is called when menu choice 3 is selected.

(2)

(e) You are required to change the solution. The phrase will not be set by the setter. Instead it will be selected at random and read from a stored file of phrases `MyPhrases.txt`. This file has one phrase per line, some of which are single words.

The file **MyPhrases.txt** is available in the **Preliminary Material** and should be accessible from your account.

What you need to do

(i) Add code to the **Skeleton Program** to implement the tasks numbered 1 to 4 which follow.

Task 1
Provide a new menu choice: `"4 . Run Question 8 code"`.
This will be used to run the new code created for the following tasks, 2, 3 and 4.

Task 2

A procedure / function `CountPhrasesFromFile`.
The procedure / function must:

- open the file `MyPhrases.txt`
- read the contents of the file
- count the number of phrases and return this number.
  (A phrase can be a single word.)

Evidence that you need to provide

PROGRAM SOURCE CODE for the procedure / function `CountPhrasesFromFile`.

(7)

(ii) Write code which calls the procedure / function `CountPhrasesFromFile` when menu choice 4 is selected.

Test that procedure / function `CountPhrasesFromFile` meets its specification.

Evidence that you need to provide

A SCREEN CAPTURE for **one** test run of the program showing:
the total number of phrases in the file.

(1)

(f) (i) **Task 3**
A procedure / function `GenerateRandomPhraseNumber`.

The procedure / function must return a random integer between 1 and *n*, say *x*, where *n* is the number of phrases in the file `MyPhrases.txt`.
Use the programming language's random number generator.

Evidence that you need to provide

PROGRAM SOURCE CODE showing
the procedure / function `GenerateRandomPhraseNumber`.

EXAM PAPERS PRACTICE

(3)

(ii) Change your code so that when menu choice 4 is selected your program calls procedures / functions:

- `CountPhrasesFromFile`
- `GenerateRandomPhraseNumber`.

Test that procedure / function `GenerateRandomPhraseNumber` meets its specification.

Evidence that you need to provide

SCREEN CAPTURES for **two** test runs of the program each showing:
x, the generated phrase number.

(2)

(g) (i) **Task 4**
A procedure / function `SelectPhraseFromFile`

The procedure / function must:

- open `MyPhrases.txt`
- read the x^{th} phrase

- return the phrase.

Test that procedure / function `SelectPhraseFromFile` meets its specification.

Evidence that you need to provide

PROGRAM SOURCE CODE showing the procedure / function `SelectPhraseFromFile`.

(7)

(ii) Change your code so that when menu choice 4 is selected:
Your program calls procedures / functions:

- `CountPhrasesFromFile`
- `GenerateRandomPhraseNumber`
- `SelectPhraseFromFile`

Then

- assigns the selected phrase to variable `NewPhrase`
- displays the output:
  `Phrase selected is: HIP HOP MUSIC`
  (or some other phrase from the file)

No attempt should be made to ask the user to guess this phrase once it has been selected.

Test that procedure / function `SelectPhraseFromFile` meets its specification.

Evidence that you need to provide

SCREEN CAPTURES for **two** test runs of the program each showing: the x^{th} phrase selected.

(2)

(h) **Evidence that you need to provide**

PROGRAM SOURCE CODE showing the declaration of any new variable(s) used in the tasks above.

(2)
(Total 33 marks)

Q23.

A binary tree has the following functions defined

`RootValue(T)`	Returns the contents of the root node of the tree T
`LeftChild(T)`	Returns the left child of the root node of the tree T
`RightChild(T)`	Returns the right child of the root node of the tree T

A recursively-defined procedure P with a tree as a parameter is defined below.

```
Procedure P(T)
  If RightChild(T) Exists
   Then P(RightChild(T))
  Output RootValue(T)
  If LeftChild(T) Exists
    Then P(LeftChild(T))
EndProc
```

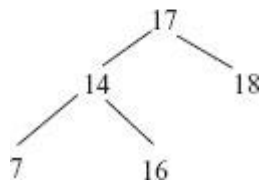
(a) What is meant by a recursively-defined procedure?

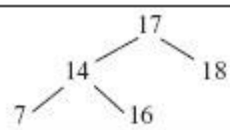
____

____

(1)

(b) (i) Complete the table below by dry running the procedure call P(T) for the tree T given below



Procedure Call	T
P_1	

EXAM TICE

Output					

(7)

(ii) What does the procedure P describe?

____

(2)
(Total 10 marks)

Q24.

A binary search tree has the following functions defined:

RootValue(T)	Returns the value stored in the root node of the tree T
LeftChild(T)	Returns the left child (subtree) of the root node of the tree T
RightChild(T)	Returns the right child (subtree) of the root node of the tree T

A recursively-defined procedure P with a tree as a parameter is defined below.

```
Procedure P(T)
  If RightChild(T) exists
    Then P(RightChild(T))
  Output RootValue(T)
  If LeftChild(T) exists
    Then P(LeftChild(T))
EndProc
```

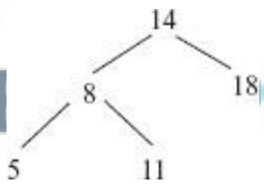
(a) What is meant by a recursively-defined procedure?

____

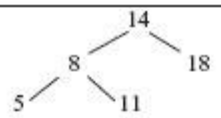
____

(1)

(b) (i) Complete the table below by dry running the procedure call P(T) for the tree T given below.



EXAM PAPERS PRACTICE

Procedure Call		Output
P_1		

(6)

EXAM PAPERS PRACTICE

(ii) What does the procedure P describe?

____

(2)
(Total 9 marks)

Q25.

(a) Writing program code requires the programmer to use identifiers for variables and procedures.

(i) State **two** other uses for identifiers.

1. ____

2. ____

(2)

(ii) Most programming languages impose restrictions or rules about what is and is not allowed for identifier names. State **one** such rule.

____

(1)

(b) Program code is often written with the use of procedures. Describe **one** reason why a programmer would decide to use procedures.

____

____

(1)

(c) A programmer-written function **SearchThisArray** is defined as follows.

```
SearchThisArray(ThisArray : Array[1..10] Of String;
                                   ThisString : String) : Integer ;

The function searches the array ThisArray for the value ThisString.

If an exact match is found, the function returns the index position
in ThisArray.
If not found, the function returns -1.
If the function's arguments, ThisArray and ThisString are illegally
formed, the
function returns -2
```

The function is used in a program with the statements shown below and uses the data shown in the `Customer` array in the figure below.

Index (Subscript)	Customer
[1]	Weeks
[2]	Adamson
[3]	Patel
[4]	Berkovic
[5]	Ince
[6]	Neale
[7]	Williamson
[8]	Collins
[9]	Davis
[10]	Beckham

What is the value returned to variable `Result` in each case?

(i) `Result := SearchThisArray(Customer, 'Beckham')`

Value of `Result` ____

(1)

(ii) `Result := SearchThisArray(Customer, 'Williams')`

Value of `Result` ____

(1)
(Total 6 marks)

Q26.

A county has a number of local libraries in various towns. Books currently belong to each library and there is no system for the exchange of books between libraries.

New programs have to be written, as the decision has been made to have centralised records of library books.

The software house commissioned to write the new programs has obtained a complete list of titles held at each library. It found that a common system was used for the book codes. Some older books will not be retained and this is to be indicated by the ToBeRetained column in the table below.

BookTitle	BookCode	YearFirstInStock	ToBeRetained
Hang-gliding made simple	T05320	1993	
Around the world in 80 days	T76542	2001	
My way	M11981	1990	
Starting with hypnotherapy	M79080	2005	
Kim Smith – the autobiography	M00876	1991	
XXX			

EXAM PAPERS PRACTICE

(a) Study the sample data shown in the table. This data will be accessed by program code. Name the most suitable **data type** which should be used for each data item. Each data type **must be different.**

(i) BookCode ____

(1)

(ii) YearFirstInStock ____

(1)

(iii) ToBeRetained ____

(1)

(b) The first application to be developed is a program to search the complete list of books and to calculate the data values for the ToBeRetained column; any books which were bought before 1992 will not be retained.

The incomplete pseudo-code which follows shows a first attempt at the algorithm. Data for each of the four attributes BookTitle, BookCode, YearFirstInStock, ToBeRetained are shown in the table above, and are to be stored in four arrays BookTitle, BookCode, YearFirstInStock and ToBeRetained.

Complete the pseudo-code in the **three** places indicated.

```
For Book ← 1 To TotalNoOfBooks
    If YearFirstInStock[ (i) ____________________ ] < 1992

       Then ToBeRetained[Book] ← (ii)____________________

       Else ToBeRetained[Book] ← (iii) ____________________
    EndIf
EndFor
```

(3)

(c) A second program is to be developed to allocate each book a new code number. The old book codes are to be abandoned. The first character of the old book code indicates the book's location.

- This book location is to be retained and stored in an array Location.
- Each new book code will be a unique integer number that will be generated by the program. The first number will be 1.

Use will be made of a 'built-in' function StartString. It is defined in the help files as follows:

Function StartString(ThisString : **String;** NoOfCharactersToRetain **: Integer) :String ;** **The function is given the string** ThisString **and returns the number of characters specified by** NoOfCharactersToRetain **starting from the first character of** ThisString.

(i) What are the values of the **parameters** used in the following code?

```
NewString : = StartString('T76542', 1)
```

1. ____

2. ____

(2)

EXAM PAPERS PRACTICE

(ii) What value is assigned to `NewString` when this code is executed?

____

(1)

(iii) The pseudo-code for the algorithm to calculate the new book codes and the locations is shown below.

```
NextAvailableCode ← 1

Book ← 1
Repeat
     If YearFirstInStock[Book] >=1992
       Then
         Begin

           LocationLetter ← StartString(BookCode[Book], 1)
           If LocationLetter = 'T'

             Then Location[Book] ← 'Torrington'
           If LocationLetter = 'M'

              Then Location[Book] ← 'Morristown'
```

```
            NewCode[Book] ⟵ NextAvailableCode

            NextAvailableCode ⟵ NextAvailableCode + 1
        End

    Book ⟵ Book + 1
Until BookTitle[Book] = 'XXX'
```

Trace the execution of this algorithm by completing the trace table **Figure 2**; use the data shown in the table **Figure 1.**

Show also the final contents of the `Location` and `NewCode` arrays in **Figure 3** and **Figure 4.**

Figure 1

	BookTitle	BookCode	YearFirstInStock
[1]	Hang-gliding made simple	T05320	1993
[2]	Around the world in 80 days	T76542	2001
[3]	My way	M11981	1990
[4]	Starting with hypnotherapy	M79080	2005
[5]	Kim Smith – the autobiography	M00876	1991
[6]	XXX		

Figure 2

NextAvailableCode	Book	LocationLetter
1	1	'T'

EXAM PAPERS PRACTICE

Figure 3 Location

[1]	
[2]	
[3]	
[4]	

Figure 4 New Code

[1]	
[2]	
[3]	
[4]	

[5] [5]

(6)
(Total 15 marks)

Q27.

(a) State the principle of operation of a set of data values which behave as a stack.

____

____

(1)

(b) Memory locations 600 to 605 are to be used as a stack area to store character data, and the first value added to the stack is to be stored at address 600.

Figure 1 shows the initial empty state of the stack.

Figure 1

600	
601	
602	
603	
604	
605	

(i) Show on **Figure 2** the state of the stack after the characters ‘A’, ‘V’, ‘E’, ‘R’ and ‘Y’ join the stack.

EXAM PAPERS PRACTICE

Figure 2

600	
601	
602	
603	
604	
605	

(1)

(ii) Two items are removed from the stack. Show on **Figure 3** the state of the stack.

Figure 3

600	
601	
602	
603	
604	
605	

(1)

(iii) Two new characters ‘S’ and ‘P’ join the stack. Show on **Figure 4** the final state of the stack.

Figure 4

600	
601	
602	
603	
604	
605	

(1)

(c) The original items in this stack are to be reversed. This can be done using a second data structure which uses locations 700 to 705 respectively. The first item added to the stack was character ‘A’.

EXAM PAPERS PRACTICE

Figure 5

Step 1 → Step 2

	Stack		(i) ______________		Stack
600	‘A’	700		600	
601	‘V’	701		601	
602	‘E’	702		602	
603	‘R’	703		603	
604	‘Y’	704		604	
605		705		605	

(before the operation) (after the operation)

(i) Name the second data structure. Label **Figure 5.**

____

(1)

(ii) Describe **Step 1** in **Figure 5.**

____

____

(1)

(iii) Describe **Step 2** in **Figure 5.**

____

____

(1)

(iv) Show on **Figure 5** the final contents of all the memory locations.

(2)

(Total 9 marks)

EXAM PAPERS PRACTICE

Q28.

A *recursively-defined* procedure **ProcA** that takes two integers as parameters is defined below.

(a) What is meant by a recursively-defined procedure?

____

____

(1)

(b) What is the role of the stack when a recursively-defined procedure is executed?

____

____

(1)

(c) Dry run the procedure call **ProcA(11,1)** using the data in the array, **Items**, by completing the trace table below.

```
Procedure ProcA (Number,Entry)
      If Number <> Items[Entry]
      Then ProcA (Number,Entry+1)
      Else Output (Entry)
      EndIf
EndProc
```

	Items
[1]	4
[2]	5
[3]	8
[4]	11
[5]	15
[6]	19

[7]	21
[8]	28
[9]	33

Number	Entry	Output
11	1	

(4)

(d) What is the purpose of this algorithm?

____

(1)

(e) Give a situation where this algorithm will fail.

____

____

(1)

(f) Suggest a modification to the algorithm that will prevent it from failing.

____

____

EXAM PAPERS PRACTICE

(1)

(g) With an ordered array, Items, of many more entries, what more efficient algorithm could be used to achieve your expressed purpose in part (d)?

____

____

(1)
(Total 10 marks)

Q29.

(a) Well constructed programs use a structured approach for the design and coding stages.

One practical way in which the programmer will use a structured approach to programming is the use of subroutines (procedures/functions). Give **three** other ways.

1. ____

2. ____

3. ____

(3)

(b) A program is to be written which calculates the hourly pay rate for an employee. The calculation is based on the number of complete years the employee has worked for the firm (e.g. 3 years). All employees get a basic £7.88 per hour. For each year worked, up to a maximum of 5 years only, an additional £0.65 is added to the basic hourly rate.

The algorithm for this program is as follows:

1. Enter the surname
2. Enter the number of years of service
3. Calculate the employee's pay rate
4. Output the surname and pay rate

(i) Complete the table showing **three** variable identifiers and their data types you would use for this problem.

Variable Identifier	Data Type

(3)

(ii) The detail for step 3 in the algorithm is broken down into more detail as follows:

3.1 If the number of years of service value is over 5, then change the value stored to 5

3.2 Calculate the employee's pay rate

Write pseudo-code for these two steps using the appropriate identifiers from the table.

3.1 ____

____

3.2 ____

____

(3)

(Total 9 marks)

Q30.

A company makes sofas and operates seven days a week. Each day a record is made of the number of sofas that are rejected at the final quality control stage. An average of one reject each day is considered acceptable. This is investigated using the program below at

EXAM PAPERS PRACTICE

the end of each week.

```
Program RejectReport;
Var
     DayNo: Integer;
     RejectTotal: Integer;
     DailyRejects: Array [1..7] of Integer;
Begin
     RejectTotal := 0;
     For DayNo := 1 To 7
          Do RejectTotal := RejectTotal + DailyRejects [DayNo];
     WriteLn(RejectTotal);
End.
```

(a) What does this program do?

____

____

____

(2)

(b) (i) Write the assignment statement in the program which performs a calculation.

____

(1)

(ii) Write a declaration statement that appears in the program.

____

(1)

(iii) What is the purpose of the variable `DayNo`?

____

(1)

EXAM PAPERS PRACTICE

(iv) What type of data structure is `DailyRejects`?

____

(1)

(c) The program is to be extended to report whether this was a satisfactory week for the number of rejected sofas. An average of one reject each day is considered acceptable.

Write additional programming statement(s), in the language you are familiar with, to report one of the messages 'Investigate' or 'Inside weekly tolerance'. Use the same variable identifiers as used in the program given.

____

____

____

____

____

____ **(2)**

(d) “*A programming team should make extensive use of program libraries*.”

Explain this statement ____

____

____

____ **(2)**

(e) Another application is to be developed. The number of rejects per week is recorded over a five-week period. This data is stored in array `NoOfRejects`. The array `WeeklySupervisor` records who the supervisor was for week 1, week 2, etc. A third array `SupervisorTotal` will record the total number of unsatisfactory weeks for each of the three supervisors.

The pseudo-code which follows in **Figure 1** makes clear which array position is used for each supervisor.

Figure 1

	NoOfRejects		WeeklySupervisor		SupervisorTotal
[5]	9	[5]	‘Jones’		
[4]	8	[4]	‘Summers’		
[3]	1	[3]	‘Jones’	[3]	
[2]	9	[2]	‘Summers’	[2]	
[1]	8	[1]	‘Franks’	[1]	

```
SupervisorTotal [1] ←0

SupervisorTotal [2] ←0

SupervisorTotal [3] ←0

For WeekNo ←1 to 5

     ThisNumber←NoOfRejects [WeekNo]
     If ThisNumber > 7 Then
          Output ‘Investigate’
          Call AddToSupervisorTotal
     End If
End For
Procedure AddToSupervisorTotal
If WeeklySupervisor [WeekNo] = ‘Franks’

     Then SupervisorTotal [1] ←SupervisorTotal [1] + 1
End If
If WeeklySupervisor [WeekNo] = ‘Summers’
```

EXAM PAPERS PRACTICE

```
     Then SupervisorTotal [2] ←SupervisorTotal [2] + 1
End If
If WeeklySupervisor [WeekNo] = 'Jones'
     Then SupervisorTotal [3] ←SupervisorTotal [3] + 1
End If
End Procedure
```

(i) The number of unsatisfactory weeks when Jones was in charge is stored in the array `SupervisorTotal`. At what position in the array is this number stored?

____

(1)

(ii) Trace the algorithm by completing the trace table in **Table 1.**

Table 1

WeekNo	ThisNumber	Output	SupervisorTotal [1]	[2]	[3]
			0	0	0
1					

EXAM PAPERS PRACTICE

(6)
(Total 17 marks)

Q31.

A retail store employs ten sales staff. Staff try to persuade customers to take out a store card with the company when they make a purchase. The store keeps a record of the number of new store cards issued by its sales staff over the first six months of the year.

Table 1

StoreCards

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	12	12	6	8	3	2
[2]	12	17	7	4	5	6
[3]	2	12	0	12		

[4]	4	10	7	4		
[5]	5	0	0	0	0	0
[6]	6	1	4	6	7	8
[7]	12	19	12	**16**	17	6
[8]	13	9	7	3	4	5
[9]	12	8	4	4	5	4
[10]	14	11	12	4	5	6

The data is to be stored in a 2-dimensional array with identifier StoreCards as shown in the table above The first subscript of the array represents the row number (the salesperson number), and the second subscript the column number (the month).

(a) In the table the value 16 has been **emboldened**. Explain what this value represents.

____ **(2)**

(b) Write a declaration statement for the array StoreCards.

____ **(2)**

(c) Using the data given in the table abobe, write an assignment statement for the January sales for salesperson 8.

____ **(2)**

EXAM PAPERS PRACTICE

(d) Study the pseudo-code below.

```
Input SalesPersonNumber
PersonTotal ← 0
For Month ← 1 to 6 Do
    PersonTotal ← PersonTotal +
    storeCards[SalesPersonNumber, Month]
End For
Print PersonTotal
```

Explain what this algorithm is designed to do.

____

____

____

____ **(2)**

(e) A number of programs are to be written for the store card application, and the following are some of the data values which will need to be stored and/or calculated.

State what data type the programmer would use for each data item below.

(i) Average overtime hours worked by each member of staff.

____

(1)

(ii) Whether or not the staff are willing to work on Boxing Day.

____

(1)

(iii) The number of customer complaints made about each member of staff.

____

(1)

(Total 11 marks)

Q32.

(a) (i) Explain **one** difference between a procedure and a function.

____

____

____

(2)

(ii) Name and describe a built-in function you have used in your programming work, or when using a generic software package.

EXAM PAPERS PRACTICE

____

____

(2)

(b) A particular built-in function is described in a programming language's help files as follows:

> Function MatchString(ThisString , StringSearchedFor : String) :Boolean
>
> The function **MatchString** returns a **Boolean** value indicating whether or not the string **StringSearchedFor** appears within the string **ThisString**.
>
> An error is returned when a function call is incorrectly formed.

What value is returned to the Result1, Result2 and Result3 variables from the following function calls?

(i) Result1 := MatchString ('Harry Potter', 'Pot')

____

(1)

(ii) Result2 := MatchString (‘Potter’, ‘Harry Potter’)

____

(1)

(iii) Result3 := MatchString (‘Harry Potter’, 59)

____

(1)

(c) In part (b) (i) Result1 is an identifier used for a variable. Name **two** other uses for identifiers in a high level language.

1. ____

2. ____

(2)

(d) The programming language being used has both compiler and interpreter software for program development.

Give **one** advantage of the use of each.

Interpreter advantage ____

____

____

(1)

Compiler advantage ____

____

EXAM PAPERS PRACTICE

____

(1)

(Total 11 marks)

Q33.

The data shown below is a list of surnames of 20 motor car policyholders with the number of claims they have each made in the last five years.

	PolicyHolder		NoOfClaims
1	Wilcox	1	1
2	Adams	2	0
3	Pollard	3	0
4	Williams	4	0
5	Searle	5	3

6	Kelly	6	0
7	Lewis	7	1
8	Franks	8	5
9	Patel	9	1
10	Li Che	10	0
...		...	...
...		...	...
19	Wilkinson	19	3
20	Veale	20	0

(a) (i) The user inputs a policyholder. If the surname is found, the program outputs the number of claims for that policyholder.

```
     Read(SearchName)
     For P := 1 To 20 Do
            If PolicyHolder[P] = SearchName
               Then GoTo 200
     GoTo 300
200 : Write(NoOfClaims[P])
300: End
```

Give **two** reasons why this is badly designed program code.

1. ____

____

2. ____

____

(2)

(ii) Write declaration statements (in a language with which you are familiar) for the PolicyHolder or NoOfClaims data structure above, and one other variable used in the code above.

The programming language I am using is ______________________________

1. ____

2. ____

(2)

(b) A new task is to design and write code to establish if there are any policyholders who have made five or more claims. The program will output a ‘yes’ or ‘no’ message only.

Write the code for this new task in a programming language with which you are familiar.

EXAM PAPERS PRACTICE

(Hint: Use a loop structure to initiate the loop, and then end the loop when some condition is met.)

(5)
(Total 9 marks)

Q34.

A stack may be implemented by using either an array or a linked list.

(a) Give a disadvantage of:

(i) an array implementation;

(1)

(ii) a linked list implementation.

(1)

(b) Under what circumstances would it be more appropriate to use:

(i) an array;

(1)

(ii) a linked list.

EXAM PAPERS PRACTICE

____

(1)

(Total 4 marks)

Q35.

A tree has the following functions defined:

RootValue(T)	Returns the contents of the root node of the tree T
LeftChild(T)	Returns the left child of the root node of the tree T
RightChild(T)	Returns the right child of the root node of the tree T

A recursively-defined procedure P with a tree as a parameter is defined below.

```
Procedure P (T)
     If LeftChild(T) exists
          then P(LeftChild(T))
     Output RootValue(T)
     If RightChild(T) exists
          then P(RightChild(T))
EndProc
```

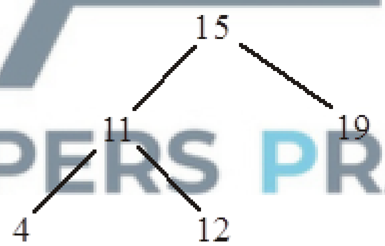
(a) What is meant by recursively-defined?

____

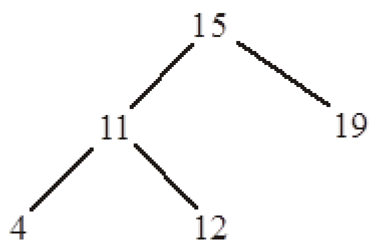
____

(1)

(b) (i) Complete the table below by dry running the procedure call P(T) for the tree T given below.



EXAM PAPERS PRACTICE

Procedure Call	T		Output
P_1			

(6)

(ii) What does procedure P describe?

____

(2)
(Total 9 marks)