Version 1.0 0711

General Certificate of Education (A-level)
June 2011

Computing COMP1

(Specification 2510)

Unit 1: Problem Solving, Programming, Data Representation and Practical Exercise

Final

Mark Scheme

Mark schemes are prepared by the Principal Examiner and considered, together with the relevant questions, by a panel of subject teachers. This mark scheme includes any amendments made at the standardisation events which all examiners participate in and is the scheme which was used by them in this examination. The standardisation process ensures that the mark scheme covers the candidates' responses to questions and that every examiner understands and applies it in the same correct way. As preparation for standardisation each examiner analyses a number of candidates' scripts: alternative answers not already covered by the mark scheme are discussed and legislated for. If, after the standardisation process, examiners encounter unusual answers which have not been raised they are required to refer these to the Principal Examiner.

It must be stressed that a mark scheme is a working document, in many cases further developed and expanded on the basis of candidates' reactions to a particular paper. Assumptions about future mark schemes on the basis of one year's document should be avoided; whilst the guiding principles of assessment remain constant, details will change, depending on the content of a particular examination paper.

Further copies of this Mark Scheme are available from: aqa.org.uk

Set and published by the Assessment and Qualifications Alliance.

The Assessment and Qualifications Alliance (AQA) is a company limited by guarantee registered in England and Wales (company number 3644723) and a registered charity (registered charity number 1073334).
Registered address: AQA, Devas Street, Manchester M15 6EX.

To Examiners:

When to award '0' (zero) when inputting marks on QMS *and on scripts*: A mark of 0 should be awarded where a candidate has attempted a question but failed to write anything creditworthy. Insert a hyphen when a candidate has not attempted a question. By these two actions the Principal Examiner will be able to distinguish between the two (nothing credit worthy/unattempted) when analysing any statistics.

The following annotation is used in the mark scheme:

; - means a single mark
// - means alternative response
/ - means an alternative word or sub-phrase
A - means acceptable creditworthy answer
R - means reject answer as not creditworthy
NE - means not enough
I - means ignore

No marks will be awarded for answers to testing questions where there is no evidence of programming code for the question(s) asked or where the screen captures provided by the candidate do not match what would be produced by the programming code provided by the candidate.

Qu	Part	Marking Guidance	Marks
1	**01**	0111 1011;	**1**
	02	256 // 2^8;	**1**
	03	7;B;	**2**
	04	Easier for <u>people</u> to read/understand; (Can be displayed using) fewer digits; More compact when printed/displayed; **NE.** Takes up less space **NE.** More compact	**Max 1**

2	**05**	011;	**1**
	06	010;	**1**
	07	110;	**1**
	08	Gray code counters consume half the/less electrical power; Prevents some errors that can happen when the value changes; (When a value is incremented only one bit changes at a time therefore) there is less likelihood of an error occurring; **A.** Fewer errors	**Max 1**

3	**09**	The number of pixels/dots; per cm/inch/unit of measurement;	**2**
	10	The number of bits used to represent (the colour/greyscale value); **R.** number of (different) colours of a single pixel;	**2**
	11	50;;// 10*10;*4÷8;//100; ÷2;//100;*0.5; **MAX 1** if final answer not correct	**2**
	12	Does not deteriorate (**A.** Concept of deteriorating by implication) when enlarged/magnified // (usually) faster to transmit // (usually) faster to load // (usually) uses less memory/storage space // Easier to edit/manipulate objects in the image (**A.** Alternative word to object); **NE.** Easier to edit/manipulate	**1**

4	**13**	Design;	**1**
	14	Testing; **A.** Installation	**1**

5	**15**	To measure out one litre of water;	**1**
	16	3 litre capacity jug; and 5 litre capacity jug; **A.** 2 jugs; sink; with a tap/water; drain; Bob // Bob's knowledge and problem solving skills; Time;	**Max 3**
	17	Who is responsible for solving the problem;	**1**

6	**18**	18, 23, 21, 36, 40, 45, 58, 59 **Mark as follows:** 18 in the first place; 23 and 21 in correct order and in the second and third places; 21 and 36 in the correct order and in the third and fourth places; 40, 45, 58 and 59 in the correct order and in the last four places; **A.** Table 3 instead of Table 2 as long as the bottom cell of each of the scores column is correct (**I.** any working out)	**4**
	19	Bubble sort; **NE.** sort	**1**

7	20	**VB.Net**
		`Sub Main()`
		`  Dim Names(4) As String`
		`  Dim Current As Integer`
		`  Dim Max As Integer`
		`  Dim Found As Boolean`
		`  Dim PlayerName As String`
		
		`  Names(1) = "Ben"`
		`  Names(2) = "Thor"`
		`  Names(3) = "Zoe"`
		`  Names(4) = "Kate"`
		`  Max = 4`
		`  Current = 1`
		`  Found = False`
		`  Console.WriteLine("What player are you looking for?")`
		`  PlayerName = Console.ReadLine`
		`  While Found = False And Current <= Max`
		`  If Names(Current) = PlayerName Then`
		`  Found = True`
		`  Else`
		`  Current = Current + 1`
		`  End If`
		`  End While`
		`  If Found = True Then`
		`  Console.WriteLine("Yes, they have a top score")`
		`  Else`
		`  Console.WriteLine("No, they do not have a top score")`
		`  End If`
		`  Console.ReadLine()`
		`End Sub`
		
		VB6
		`Private Sub Form_Load()`
		`  Dim Names(4) As String`
		`  Dim Current As Integer`
		`  Dim Max As Integer`
		`  Dim Found As Boolean`
		`  Dim PlayerName As String`
		
		`  Names(1) = "Ben"`
		`  Names(2) = "Thor"`
		`  Names(3) = "Zoe"`
		`  Names(4) = "Kate"`
		`  Max = 4`
		`  Current = 1`
		`  Found = False`
		`  PlayerName = InputBox("What player are you looking for?")`
		`  While Found = False And Current <= Max`
		`  If Names(Current) = PlayerName Then`
		`  Found = True`
		`  Else`
		`  Current = Current + 1`
		`  End If`
		`  End While`
		`  If Found = True Then`
		`  MsgBox("Yes, they have a top score")`
		`  Else`
		A. `Names(1 To 4)`	

```
    MsgBox("No, they do not have a top score")
  End If
  End
End Sub
```

Pascal

```
Program Question7;
Var
  Names : Array[1..4] Of String;
  Current : Integer;
  Max : Integer;
  Found : Boolean;
  PlayerName : String;
Begin
  Names[1] := 'Ben';
  Names[2] := 'Thor';
  Names[3] := 'Zoe';
  Names[4] := 'Kate';
  Max := 4;
  Current := 1;
  Found := False;
  Writeln('What player are you looking for?');
  Readln(PlayerName);
  While (Found = False) And (Current <= Max)
    Do
      Begin
        If Names[Current] = PlayerName
          Then Found := True
          Else Current := Current + 1;
      End;
  If Found = True
    Then Writeln('Yes, they have a top score')
    Else Writeln('No, they do not have a top
score');
  Readln;
End.
```

Mark as follows:

Correct variable declarations for `Max`, `Current`, `Found`, `PlayerName` and correct declaration for the `Names` array;
Four correct values assigned to the correct positions in the `Names` array;
`Max`, `Current`, `Found` initialised correctly;
Correct prompt followed by `PlayerName` assigned value entered by user;
WHILE loop formed correctly and correct conditions for the termination of the loop;
First IF followed by correct condition and IF statement is inside the loop;
THEN followed by correct assignment statement within a correctly formed IF statement;
ELSE followed by correct assignment statement within a correctly formed IF statement;
Second IF followed by correct condition and IF is after the loop;
THEN followed by correct output within a correctly formed IF statement;
ELSE followed by correct output within a correctly formed IF statement;

I. Case of variable names, player names and output messages

		A. Minor typos in variable names and output messages **A.** `Max` declared as a constant instead of a variable **A.** Alternative conditions with equivalent logic for the loop **A.** Array positions 0-3 used instead of 1-4 if consistent usage throughout program	**11**
	21	****SCREEN CAPTURE**** *Must match code from 20, including prompts on screen capture matching those in code. Code for 20 must be sensible.* **Mark as follows:** 'What player are you looking for?' + user input of 'Thor'; 'Yes, they have a top score' message shown; **I.** spacing **R.** If code for 20 would not produce this test run	**2**
	22	****SCREEN CAPTURE**** *Must match code from 20, including prompts on screen capture matching those in code. Code for 20 must be sensible.* **Mark as follows:** 'What player are you looking for?' + user input of 'Imran'; 'No, they do not have a top score' message shown; **I.** spacing **R.** If code for 20 would not produce this test run	**2**

8	**23**	**VB.Net/VB6**
		`Const MaxSize = 4`
		I. capitalisation
		
		Pascal
		`Const MaxSize = 4;`
		I. missing semicolon, capitalisation
		
		NE. MaxSize
		A. MaxSize = 4
			1

	24	Improves readability of code // Easier to update the programming code if the value changes (**A.** by implication) // reduce the likelihood of errors;	**1**
	25	`PlayerOneName` // `PlayerTwoName`; **R.** if any additional code **R.** if spelt incorrectly **I.** case & spaces **A.** `MAX_SIZE` (Python only) **A.** `Currentfile` (**R.** for VB6/VB.Net)	**1**
	26	`LowestCurrentTopScore` ; **A.** `PositionOfLowestCurrentTopScore`; **R.** if any additional code **R.** if spelt incorrectly **I.** case & spaces	**1**
	27	b;	**1**

	28	`True;`	**1**
	29	`False;`	**1**
	30	`UpdateTopScores`; **R.** if spelt incorrectly **I.** case & spaces	**1**
	31	`VirtualDiceGame;` **R.** if spelt incorrectly **I.** case & spaces	**1**

	32	`AppealDieResult;`
		`RollAppealDie;`
		
		R. if spelt incorrectly
		R. `RollAppealDie` (Python only)
		I. case & spaces
			1

	33	Until PlayerOut // Until PlayerOut = True // until player is out; **A.** any unambiguous description of the loop termination condition	**1**
	34	Because the scope; of the two variables is different; // Because they are both local variables; in different subroutines; **A.** Because where they are accessible is different;;	**2**
	35	3;	**1**
	36	It compares the score of the current record/position (in the `TopScores` array); with the lowest score found so far // with LowestCurrentTopScore; if it is less than it then it changes the lowest score found so far; **R.** swaps and makes the position of the lowest top score equal to count / equal to the current position in the array;	**4**

9	37	**VB.Net**
		`If VirtualDiceGame Then`
		`  AppealDieResult = Int(Rnd() * 5) + 1`
		`Else`
		`  Console.WriteLine("Please roll the appeal die and then enter your result.")`
		`  Console.WriteLine()`
		`  Console.WriteLine("Enter 1 if the result is NOT OUT")`
		`  Console.WriteLine("Enter 2 if the result is CAUGHT")`
		`  Console.WriteLine("Enter 3 if the result is LBW")`
		`  Console.WriteLine("Enter 4 if the result is BOWLED")`
		`  **Console.WriteLine("Enter 5 if the result is RUN OUT")**`
		`  Console.WriteLine()`
		`  Console.Write("Result: ")`
		`  AppealDieResult = Console.ReadLine`
		`  Console.WriteLine()`
		`End If`
		
		VB6
		`  If VirtualDiceGame Then`
		`    AppealDieResult = Int(Rnd() * 5) + 1`
		`  Else`
		`    WriteLine ("Please roll the appeal die and then enter your result.")`
		`    WriteLine ("")`
		`    WriteLine ("Enter 1 if the result is NOT OUT")`
		`    WriteLine ("Enter 2 if the result is CAUGHT")`
		`    WriteLine ("Enter 3 if the result is LBW")`
		`    WriteLine ("Enter 4 if the result is BOWLED")`
		`    **WriteLine ("Enter 5 if the result is RUN OUT")**`
		`    WriteLine ("")`
		`    AppealDieResult = ReadLine("Result:")`
		`    WriteLine ("")`
		`  End If`
		
		A. Text1.Text = Text1.Text & "Enter 5 if the result is RUN OUT"
		A. WriteLineWithMsg
		
		Pascal
		`If VirtualDiceGame`
		`  Then AppealDieResult := Random(5) + 1`
		`  Else`
		`    Begin`
		`      Writeln('Please roll the appeal die and then enter your result.');`
		`      Writeln;`
		`      Writeln('Enter 1 if the result is NOT OUT');`
		`      Writeln('Enter 2 if the result is CAUGHT');`
		`      Writeln('Enter 3 if the result is LBW');`
		`      Writeln('Enter 4 if the result is BOWLED');`
		`      **Writeln('Enter 5 if the result is RUN OUT');**`
		`      Writeln;`
		`      Write('Result: ');`
		`      Readln(AppealDieResult);`
		`      Writeln;`
		`    End;`	

Mark as follows:
Generates random number between 1 and 5;
Appropriate prompt added if real dice being used;
I. minor typos and capitalisation in prompt
A. alternative sensible prompt

2

38

VB.Net

```
Select Case AppealDieResult
  Case 1
    Console.WriteLine("Not out!")
  Case 2
    Console.WriteLine("Caught!")
  Case 3
    Console.WriteLine("LBW!")
  Case 4
    Console.WriteLine("Bowled!")
  Case 5
    Console.WriteLine("Run Out!")
End Select
```

VB6

```
Select Case AppealDieResult
  Case 1
    WriteLineWithMsg ("Not out!")
  Case 2
    WriteLineWithMsg ("Caught!")
  Case 3
    WriteLineWithMsg ("LBW!")
  Case 4
    WriteLineWithMsg ("Bowled!")
  Case 5
    WriteLineWithMsg ("Run out!")
End Select
```

A. WriteLine/WriteWithMsg/Msgbox instead of WriteLineWithMsg
A. Text1.Text = Text1.Text & "Run out!"

Pascal

```
Case AppealDieResult Of
      1 : Writeln('Not out!');
      2 : Writeln('Caught!');
      3 : Writeln('LBW!');
      4 : Writeln('Bowled!');
      5 : Writeln('Run out!');
    End;
```

Mark as follows:
5th case option added;
Appropriate output message in 5th case option;
I. minor typos and capitalisation in output message

2

	39	****SCREEN CAPTURE(S)**** *This is conditional on sensible code for 37 and 38* Screen capture showing run out (option 5) message shown to user; User enters "5" and correct output message showing 'RUN OUT!'; **A.** Alternative output message if matches code for 37/38	**2**

10	**40**	**VB.Net**
		`If PlayerOneScore > PlayerTwoScore Then`
		`Console.WriteLine(PlayerOneName & " wins!")`
		`If PlayerTwoScore > PlayerOneScore Then`
		`Console.WriteLine(PlayerTwoName & " wins!")`
		`If PlayerOneScore = PlayerTwoScore Then`
		`Console.WriteLine("A draw!")`
		
		VB6
		`If PlayerOneScore > PlayerTwoScore Then`
		`WriteLineWithMsg (PlayerOneName & " wins!")`
		`If PlayerTwoScore > PlayerOneScore Then`
		`WriteLineWithMsg (PlayerTwoName & " wins!")`
		`If PlayerOneScore = PlayerTwoScore Then`
		`WriteLineWithMsg ("A draw!")`
		
		A. Using MsgBox/WriteLine/WriteWithMsg for output instead of WriteLineWithMsg
		A. Text.Text1 = Text.Text1 & "A draw!"
		
		Pascal
		`If (PlayerOneScore > PlayerTwoScore)`
		`  Then Writeln(PlayerOneName, ' wins!');`
		`If (PlayerTwoScore > PlayerOneScore)`
		`  Then Writeln(PlayerTwoName, ' wins!');`
		`If (PlayerOneScore = PlayerTwoScore)`
		`  Then Writeln('A draw!');`
		
		Mark as follows:
		IF statement;
		with correct condition;
		suitable output message shown under, and only under, correct circumstances;
			3

	41	****SCREEN CAPTURE(S)**** **Mark as follows:** Test showing both player scores are 0; Correct message shown; *This is conditional on sensible code for 40*	**2**

11	42	**VB.Net**
		`Console.Write("Result: ")`
		`BowlDieResult = Console.ReadLine()`
		`Console.WriteLine()`
		`While BowlDieResult < 1 Or BowlDieResult > 6`
		`  Console.Writeline("Please enter a value between 1 and 6 only")`
		`  BowlDieResult = Console.ReadLine`
		`End While`
		
		Alternative Answer – VB.Net
		`Do`
		`  Console.Write("Result: ")`
		`  BowlDieResult = Console.ReadLine`
		`  If BowlDieResult < 1 Or BowlDieResult > 6 Then`
		`    Console.WriteLine("Please enter a number between 1 and 6 only")`
		`  End If`
		`Loop Until BowlDieResult >= 1 And BowlDieResult <= 6`
		
		VB6
		`BowlDieResult = ReadLine("Result:")`
		`While BowlDieResult < 1 Or BowlDieResult > 6`
		`  BowlDieResult = ReadLine("Please enter a value between 1 and 6 only")`
		`End While`
		
		A. InputBox instead of ReadLine
		
		Alternative Answer – VB6
		`Do`
		`  BowlDieResult = ReadLine("Result:")`
		`  If BowlDieResult < 1 Or BowlDieResult > 6 Then`
		`    BowlDieResult = WriteLine("Please enter a value between 1 and 6 only")`
		`  End If`
		`Loop Until BowlDieResult >= 1 And BowlDieResult <= 6`
		
		Pascal
		`Repeat`
		`  Write('Result: ');`
		`  Readln(BowlDieResult);`
		`  If (BowlDieResult < 1) Or (BowlDieResult > 6)`
		`    Then Writeln('Please enter a value between 1 and 6 only');`
		`  Until (BowlDieResult >= 1) And (BowlDieResult <=6);`
		
		Alternative Answer - Pascal
		`Write('Result: ');`
		`Readln(BowlDieResult);`
		`Writeln;`
		`While (BowlDieResult < 1) Or (BowlDieResult > 6)`
		`  Do`
		`    Begin`
		`      Writeln('Please enter a value between 1 and 6 only');`	

		`Readln(BowlDieResult);`
		`End;`
		
		Mark as follows:
		Suitable iteration structure used in appropriate place in the Skeleton Program with one correct condition;
		Use of OR logical operator and have second condition correct for iterative structure;
		A. Alternative logic using AND and NOT logical operators
		Correct error message and get choice from user – both inside the loop;
		Error message is displayed if, and only if, invalid data entered by user;
		
		I. minor typos and capitalisation in output message
			4

	43	****SCREEN CAPTURE(S)**** *This is conditional on sensible code for 42* **Mark as follows:** Test showing a value of 0 entered and the correct output message; Test showing a value of 2 entered and the correct output message; Test showing a value of 7 entered and the correct output message; **I.** Order of tests **A.** Alternative error message if matches code for 42	**3**

12	**44**	**VB.Net**
		`Console.WriteLine("4.  Display top scores")`
		`Console.WriteLine("5.  Save top scores")`
		`Console.WriteLine("9.  Quit")`
		
		VB6
		`WriteLine ("4.  Display top scores")`
		`WriteLine ("5.  Save top scores")`
		`WriteLine ("9.  Quit")`
		
		Pascal
		`Writeln('4.  Display top scores');`
		`Writeln('5.  Save top scores');`
		`Writeln('9.  Quit');`
		
		A. minor typos in output message
			1

	45	**VB.Net / VB6**
		`If OptionChosen < 1 Or (OptionChosen > 5 And OptionChosen <> 9) Then`
		
		Pascal
		`If (OptionChosen < 1) Or ((OptionChosen > 5) And (OptionChosen <> 9))`
		`  Then`
		
		Mark as follows:
		OptionChosen > 5 // OptionChosen >= 6;
			1

46

VB.Net

```
Sub SaveTopScores(ByVal TopScores() As TTopScore)
  Dim Count As Integer
  Dim LineToAddToFile As String
  FileOpen(1, "HiScores.txt", OpenMode.Output)
  For Count = 1 To MaxSize
    LineToAddToFile = TopScores(Count).Name & "," &
TopScores(Count).Score
    PrintLine(1, LineToAddToFile)
  Next
  FileClose(1)
End Sub
```

VB6

```
Private Sub SaveTopScores(ByRef TopScores() As
TTopScore)
Dim Count As Integer
Open "HiScores.txt" For Output As #1
For Count = 1 To MaxSize
  Print #1, TopScores(Count).Name & "," &
Str(TopScores(Count).Score)
Next
Close #1
End Sub
```

Pascal

```
Procedure SaveTopScores(TopScores : TTopScores);
Var
  Count : Integer;
  LineToAddToFile : String;
  CurrentFile : TextFile;
Begin
  Assign(CurrentFile, 'HiScores.txt');
  ReWrite(CurrentFile);
  For Count := 1 To MaxSize
    Do
      Begin

        LineToAddToFile :=
IntToStr(TopScores[Count].Score)
        LineToAddToFile := TopScores[Count].Name +
',' + LineToAddToFile;
        Writeln(CurrentFile, LineToAddToFile);
      End;
  Close(CurrentFile);
End;
```

A. `Str(TopScores[Count].Score, LineToAddToFile);` instead of `LineToAddToFile := IntToStr(TopScores[Count].Score)`

Mark as follows:
Correctly named subroutine declared; **I.** capitalisation **R.** other mistakes in identifier
File opened correctly (for output);
First line to add into file consists of the 1st name; a comma and the 1st score;
First line written to file correctly;
2nd, 3rd and 4th lines would be written to the file correctly;

		File closed correctly; **Additional marks for good programming practice (Max 3):** `TopScores` array passed as a parameter; Use of iterative structure and counter used within iterative structure - going from 1 to MaxSize (**R. 4**); Sensible identifier names used for <u>all</u> variables/parameters; Evidence of sensible commenting of source code;	**Max 10**

	47	**VB.Net**
		`Loop Until (OptionSelected >= 1 And OptionSelected <= 5) Or OptionSelected = 9`
		`Console.WriteLine()`
		`If OptionSelected >= 1 And OptionSelected <= 5 Then`
		`  Select Case OptionSelected`
		`    Case 1 : PlayDiceGame(PlayerOneName, PlayerTwoName, True, TopScores)`
		`    Case 2 : PlayDiceGame(PlayerOneName, PlayerTwoName, False, TopScores)`
		`    Case 3 : LoadTopScores(TopScores)`
		`    Case 4 : DisplayTopScores(TopScores)`
		`    Case 5 : SaveTopScores(TopScores)`
		`  End Select`
		
		VB6
		`Loop Until (OptionSelected >= 1 And OptionSelected <= 5) Or OptionSelected = 9`
		`If OptionSelected >= 1 And OptionSelected <= 5 Then`
		`  Select Case OptionSelected`
		`    Case 1: Call PlayDiceGame(PlayerOneName, PlayerTwoName, True, TopScores)`
		`    Case 2: Call PlayDiceGame(PlayerOneName, PlayerTwoName, False, TopScores)`
		`    Case 3: LoadTopScores(TopScores)`
		`    Case 4: Call DisplayTopScores(TopScores)`
		`    Case 5: Call SaveTopScores(TopScores)`
		
		Pascal
		`Until OptionSelected In [1..5, 9];`
		`Writeln;`
		`If OptionSelected In [1..5]`
		`  Then`
		`    Case OptionSelected Of`
		`      1 : PlayDiceGame(PlayerOneName, PlayerTwoName, True, TopScores);`
		`      2 : PlayDiceGame(PlayerOneName, PlayerTwoName, False, TopScores);`
		`      3 : LoadTopScores(TopScores);`
		`      4 : DisplayTopScores(TopScores);`
		`      5 : SaveTopScores(TopScores);`
		`End;`
		
		Mark as follows:
		Additional case statement for `OptionSelected` being 5;
		Procedure call;
		Passing `TopScores` as a parameter;
		Loop terminating condition **and** selection condition range both changed from 1-4 to 1-5;
			4

	48	****SCREEN CAPTURE**** Adapted menu is displayed; *This is conditional on sensible answer for question 44* option 5 is selected, and accepted as valid input; *This is conditional on sensible answer for questions 45 and 47*	**2**
	49	****SCREEN CAPTURE**** *This is conditional on sensible answer for 45, 46 and 47* Contents of file are **exactly** as follows: Ricky,12 Sachin,45 Brian,2 Janet,4 **A.** Screen capture showing contents of text file **I.** Minor typos & capitalisation in Janet's name **R.** If Janet's name in the text file does not match the name used in 48	**1**

13	**50**	Generate wider range of random numbers; add extra case statements for low score values / give low score values a bigger range in case statements than high score values; // Create a list/array containing a list of possible bowl die results where there are more 1s and 5s than 3s and 4s; generate a random number between 1 and the list size and use the bowl die result in that position in the list/array; **Mark as follows:** Generate a wider range of random numbers; Explain how the extra random numbers could be used to have a higher chance of getting a score of 1 or 0 than a score of 4 or 6; **A.** Replace case statement with if statements to allow different score values to have ranges of values associated with them **(Pascal Only)** **A.** Other sensible suggestions for modifications to the Skeleton Program that would result in the desired behaviour change **MAX 1** if suggested changes would adversely effect other aspects of the game represented in the Skeleton Program e.g. does result in more lower scores than higher scores but would prevent a player from getting a result of out.	**2**

C Mark Scheme

7	**20**	`char* Names[5];`
		`int Current;`
		`int Max;`
		`int Found;`
		`char* PlayerName;`
		
		`void main(void)`
		`{`
		`  Names[1] = "Ben";`
		`  Names[2] = "Thor";`
		`  Names[3] = "Zoe";`
		`  Names[4] = "Kate";`
		`  Max = 4;`
		`  Current = 1;`
		`  Found = 0;`
		`  printf("%s", "What player are you looking for? ");`
		`  fgets(PlayerName);`
		`  while (!Found && Current <= Max)`
		`  {`
		`  if (strcmp(Names[Current],PlayerName) == 0)`
		`  Found = 1;`
		`  else`
		`  Current++;`
		`  };`
		`  if (Found)`
		`  print("%s", "Yes, they have a top score\n");`
		`  else`
		`  printf("%s", "No, they do not have a top score\n");`
		`  printf("%s", "Press the Enter key to continue\n");`
		`  _getch();`
		`}`
			11

8	**23**	`#define MaxSize 4`	**1**

9	**37**	`if (VirtualDiceGame)`
		`{`
		`  AppealDieResult = random(5) + 1;`
		`}`
		`else`
		`{`
		`  printf("%s","Please roll the appeal die and then enter your result.\n");`
		`  printf("%s", "Enter 1 if the result is BOWLED\n");`
		`  printf("%s", "Enter 2 if the result is CAUGHT\n");`
		`  printf("%s", "Enter 3 if the result is LBW\n");`
		`  printf("%s", "Enter 4 if the result is NOT OUT\n");`
		`  **printf("%s", "Enter 5 if the result is RUN OUT\n");**`
		`  scanf("%d", &AppealDieResult);`
		`  printf("%s", "\n");`
		`}`
			2

	38	`switch (AppealDieResult)`
		`{`
		`  case 1: printf("%s", "Bowled!\n");`
		`  break;`
		`  case 2: printf("%s", "Caught!\n");`
		`  break;`
		`  case 3: printf("%s", "LBW!\n");`
		`  break;`
		`  case 4: printf("%s", "Not out!\n");`
		`  break;`
		`  **case 5: printf("%s", "Run out!\n");**`
		`  **break;**`
		`}`
			2

	40	`if (PlayerOneScore > PlayerTwoScore)`
		`  printf("%s%s", PlayerOneName, " wins!\n");`
		`if (PlayerTwoScore > PlayerOneScore)`
		`  printf("%s%s", PlayerTwoName, " wins!\n);`
		`if (PlayerOneScore == PlayerTwoScore)`
		`  printf("%s", "A draw!\n");`
			3

	42	`printf("%s", "Result: ");`
		`scanf("%d", &BowlDieResult);`
		
		`while (BowlDieResult < 1 || BowlDieResult > 6)`
		`{`
		`  printf("%s", "Please enter a value between 1 and 6 only\n");`
		`  scanf("%d", &BowlDieResult);`
		`}`
			4

	44	`printf("%s", "4. Display top scores\n");`
		`printf("%s", "5. Save top scores\n");`
		`printf("%s", "9. Quit\n");`
			1

	45	`if ((OptionChosen < 1 || (OptionChosen > 5) && OptionChosen != 9)`	**1**

	46	`void SaveTopScores(struct TTopScore TopScores[5])`
		`{`
		`  int Count;`
		`  char Buf[5];`
		`  char LineToAddToFile[255];`
		`  char TempString[255];`
		`  FILE* CurrentFile;`
		
		`  CurrentFile = fopen("HiScores.txt", "w");`
		`  for (Count = 1; Count <= MaxSize; Count++)`
		`  {`
		`  strcpy(TempString, TopScores[Count].Name);`
		`  strcat(TempString, ",");`
		`  strcpy(LineToAddToFile, TempString);`
		`  itoa(TopScores[Count].Score, Buf, 10);`
		`  strcat(LineToAddToFile, Buf);`
		`  strcat(LineToAddToFile, "\n");`
		`  fputs(LineToAddToFile, CurrentFile);`
		`  }`
		`  fclose(CurrentFile);`
		`}`
			10

	47	`} while (!(OptionSelected >= 1 && OptionSelected <= 5 || OptionSelected == 9))`
		`printf("%s", "\n");`
		`if (OptionSelected >= 1 && OptionSelected <= 5)`
		`{`
		`  switch (OptionSelected)`
		`{`
		`  case 1: PlayDiceGame(PlayerOneName, PlayerTwoName, true, TopScores);`
		`  break;`
		`  case 2: PlayDiceGame(PlayerOneName, PlayerTwoName, false, TopScores);`
		`  break;`
		`  case 3: LoadTopScores(TopScores);`
		`  break;`
		`  case 4: DisplayTopScores(TopScores);`
		`  break;`
		`  **case 5: SaveTopScores(TopScores);**`
		`  break;`
		`}`
		`}`
			4

C# Mark Scheme

7	**20**	`namespace Question7`
		`{`
		`  class Program`
		`  {`
		`  public static string[] Names = new string[5];`
		`  public static int Current;`
		`  public static int Max;`
		`  public static bool Found;`
		`  public static string PlayerName;`
		
		`  static void Main(string[] args)`
		`  {`
		`  Names[1] = "Ben";`
		`  Names[2] = "Thor";`
		`  Names[3] = "Zoe";`
		`  Names[4] = "Kate";`
		`  Max = 4;`
		`  Current = 1;`
		`  Found = false;`
		`  Console.WriteLine("What player are you looking for?");`
		`  PlayerName = Console.ReadLine();`
		`  while (!Found && Current <= Max)`
		`  {`
		`  if (Names[Current] == PlayerName)`
		`  Found = true;`
		`  else`
		`  Current++;`
		`  };`
		`  if (Found)`
		`  Console.WriteLine("Yes, they have a top score");`
		`  else`
		`  Console.WriteLine("No, they do not have a top score");`
		`  Console.WriteLine("Press the Enter key to continue";`
		`  Console.ReadLine();`
		`  }`
		`  }`
		`}`
		
		A. Declaring and initialising a variable in one statement
		A. Variable declarations without `Public` keyword
			11

8	**23**	`public const int MaxSize = 4` **A.** Variable declaration without `Public` keyword	**1**

9	**37**	`if (VirtualDiceGame)`
		`{`
		`  Random objRandom = new Random();`
		`  AppealDieResult = objRandom.Next(1, 5);`
		`}`
		`else`
		`{`
		`  Console.WriteLine("Please roll the appeal die and then enter your result.");`
		`  Console.WriteLine();`
		`  Console.WriteLine("Enter 1 if the result is NOT OUT");`
		`  Console.WriteLine("Enter 2 if the result is CAUGHT");`
		`  Console.WriteLine("Enter 3 if the result is LBW");`
		`  Console.WriteLine("Enter 4 if the result is BOWLED");`
		`  Console.WriteLine("Enter 5 if the result is RUN OUT");`
		`  AppealDieResult = int.Parse(Console.ReadLine());`
		`  Console.WriteLine();`
		`}`
			2

	38	`switch (AppealDieResult)`
		`{`
		`  case 1: Console.WriteLine("Not out!");`
		`  break;`
		`  case 2: Console.WriteLine("Caught!");`
		`  break;`
		`  case 3: Console.WriteLine("LBW!");`
		`  break;`
		`  case 4: Console.WriteLine("Bowled!");`
		`  break;`
		`  case 5: Console.WriteLine("Run out!");`
		`  break;`
		`}`
			2

	40	`if (PlayerOneScore > PlayerTwoScore)`
		`  Console.WriteLine(PlayerOneName + " wins!");`
		`if (PlayerTwoScore > PlayerOneScore)`
		`  Console.WriteLine(PlayerTwoName + " wins!");`
		`if (PlayerOneScore == PlayerTwoScore)`
		`  Console.WriteLine("A draw!");`
			3

	42	`Console.Write("Result");`
		`BowlDieResult = int.Parse(Console.ReadLine());`
		`while (BowlDieResult < 1 || BowlDieResult > 6)`
		`{`
		`  Console.WriteLine("Please enter a value between 1 and 6 only");`
		`  BowlDieResult = int.Parse(Console.ReadLine());`
		`}`
			4

	44	`Console.WriteLine("4. Display top scores");`
		`Console.WriteLine("5. Save top scores");`
		`Console.WriteLine("9. Quit");`
			1

	45	`if ((OptionChosen < 1 || (OptionChosen > 5) && OptionChosen != 9)`	**1**

	46	`public static void SaveTopScores(TTopScore[] TopScores)`
		`{`
		`  int Count;`
		`  string LineToAddToFile;`
		`  TextWriter CurrentFile = new StreamWriter("HiScores.txt");`
		
		`  for (Count = 1; Count <= MaxSize; Count++)`
		`  {`
		`  LineToAddToFile = TopScores[Count].Name + ",";`
		`  LineToAddToFile = LineToAddTofile + TopScores[Count].Score.ToString();`
		`  CurrentFile.WriteLine(LineToAddToFile);`
		`  }`
		`  CurrentFile.Close();`
		`}`
			10

	47	`} while (!(OptionSelected >= 1 && OptionSelected <= 5 || OptionSelected == 9))`
		`Console.WriteLine();`
		`if (OptionSelected >= 1 && OptionSelected <= 5)`
		`{`
		`  switch (OptionSelected)`
		`  {`
		`  case 1: PlayDiceGame(PlayerOneName, PlayerTwoName, true, ref TopScores);`
		`  break;`
		`  case 2: PlayDiceGame(PlayerOneName, PlayerTwoName, false, ref TopScores);`
		`  break;`
		`  case 3: LoadTopScores(ref TopScores);`
		`  break;`
		`  case 4: DisplayTopScores(TopScores);`
		`  break;`
		`  **case 5: SaveTopScores(TopScores);**`
		`  break;`
		`  }`
		`}`
			4

Java Mark Scheme

7	**20**	`public class Question7 {`
		
		`  AQAConsole console = new AQAConsole();`
		
		`  public Question7() {`
		`  String[] names = new String[5];`
		`  int max;`
		`  int current;`
		`  boolean found;`
		`  String playerName;`
		
		`  names[1] = "Ben";`
		`  names[2] = "Thor";`
		`  names[3] = "Zoe";`
		`  names[4] = "Kate";`
		
		`//possible alternative, which declares and`
		`//instantiates in one.`
		`//String[] names={"","Ben","Thor","Zoe","Kate"};`
		
		`  current = 1;`
		`  max = 4;`
		`  found = false;`
		
		`  playerName = console.readLine("What player are you looking for? ");`
		`  while ((found == false) && (current <= max)) {`
		`  if (names[current].equals(playerName)) {`
		`  found = true;`
		`  } else {`
		`  current++;`
		`  } // end if/else`
		`  } // end while`
		
		`  if (found == true) {`
		`  console.println("Yes, they have a top score");`
		`  } else {`
		`  console.println("No, they do not have a top score");`
		`  } // end if/else`
		`  }// end CONSTRUCTOR`
		
		`  /**`
		`  * @param args the command line arguments`
		`  */`
		`  public static void main(String[] args) {`
		`  new Question7();`
		`  }`
		`}`
			11

8	**23**	`final int MAX_SIZE = 4;` **I.** missing semicolon, capitalisation **NE.** `MAX_SIZE`	**1**

9	**37**	`if (virtualDiceGame) {`
		`  appealDieResult = objRandom.nextInt(5) + 1;`
		`} else {`
		`  console.println("Please roll the appeal die and then enter your result.");`
		`  console.println();`
		`  console.println("Enter 1 if the result is NOT OUT");`
		`  console.println("Enter 2 if the result is CAUGHT");`
		`  console.println("Enter 3 if the result is LBW");`
		`  console.println("Enter 4 if the result is BOWLED");`
		`  **console.println("Enter 5 if the result is RUN OUT");**`
		`  console.println();`
		`  appealDieResult = console.readInteger("Result: ");`
		`  console.println();`
		`}`
			2

	38	`switch (appealDieResult) {`
		`  case 1:`
		`  console.println("Not out!");`
		`  break;`
		`  case 2:`
		`  console.println("Caught!");`
		`  break;`
		`  case 3:`
		`  console.println("LBW!");`
		`  break;`
		`  case 4:`
		`  console.println("Bowled!");`
		`  break;`
		`  **case 5:**`
		`  **console.println("Run out!");**`
		`  **break;**  ////////////optional`
		`}`
			2

	40	`if (playerOneScore > playerTwoScore) {`
		`  console.println(playerOneName + " wins!");`
		`} // end if`
		`if (playerTwoScore > playerOneScore) {`
		`  console.println(playerTwoName + " wins!");`
		`} // end if`
		`**if (playerTwoScore == playerOneScore) {**`
		`  **console.println("A draw!");**`
		`**}**
			3

	42	`do {`
		`  bowlDieResult = console.readInteger("Result: ");`
		`  if ((bowlDieResult < 1 || bowlDieResult > 6))`
		`  {`
		`  console.println("Please enter a value between 1 and 6 only");`
		`  }`
		`} while (bowlDieResult < 1 || bowlDieResult > 6);`	

		Alternative Answer
		`bowlDieResult = console.readInteger("Result: ");`
		`while (bowlDieResult < 1 || bowlDieResult > 6){`
		`  console.println("Please enter a value between 1 and 6 only");`
		`  bowlDieResult = console.readInteger("Result: ");`
		`}`
			4

	44	`console.println("4.  Display top scores");`
		`console.println("5.  Save top scores");`
		`console.println("9.  Quit");`
			1

	45	`if ((optionChosen < 1) || ((optionChosen > 5) && (optionChosen != 9))) {`	**1**

	46	`void saveTopScores(TopScore[] topScores) {`
		`  AQAWriteTextFile currentFile = new AQAWriteTextFile();`
		`  currentFile.openFile("hitest.txt");`
		
		`  int count;`
		`  for (count = 1; count <= MAX_SIZE; count++) {`
		`  String lineToAddToFile  = topScores[count].name + ", ";`
		`  lineToAddToFile = lineToAddToFile + String.valueOf(topScores[count].score);`
		`  currentFile.writeToTextFile(lineToAddToFile);`
		`  } // end for count`
		
		`  currentFile.closeFile();`
		`}`
			10

	47	`do {`
		`  displayMenu();`
		`  optionSelected = getMenuChoice();`
		`} while (!((optionSelected >= 1 && optionSelected <= 5) || optionSelected == 9));`
		
		`if (optionSelected >= 1 && optionSelected <= 5) {`
		`  switch (optionSelected) {`
		`  case 1:`
		`  playDiceGame(playerOneName, playerTwoName, true, topScores);`
		`  break;`
		`  case 2:`
		`  playDiceGame(playerOneName, playerTwoName, false, topScores);`
		`  break;`
		`  case 3:`
		`  loadTopScores(topScores);`
		`  break;`
		`  case 4:`
		`  displayTopScores(topScores);`
		`  break;`
		`  case 5:`
		`  saveTopScores(topScores);`
		`  break;` `//optional`
		`  } // end case`
		`} // end if`
			4

PHP Mark Scheme

7	**20**	<?php
		  $Names = array()
		  $Names[1] = "Ben";
		  $Names[2] = "Thor";
		  $Names[3] = "Zoe";
		  $Names[4] = "Kate";
		  $Max = 4;
		  $Current = 1;
		  $Found = false;
		  fwrite(STDOUT, "What player are you looking for?\n");
		  $PlayerName = trim(fgets(STDIN));
		  while (!$Found && $Current <= $Max);
		  {
		  if ($Names[Current] == $PlayerName)
		  $Found = true;
		  else
		  $Current++;
		  };
		  if ($Found)
		  fwrite(STDOUT, "Yes, they have a top score\n");
		  else
		  fwrite(STDOUT, "No, they do not have a top score\n");
		  fgets(STDIN);
		?>
			11

8	**23**	define("MaxSize", 4);	**1**
	25	$PlayerOneName / $PlayerTwoName	**1**
	26	$LowestCurrentTopScore / $PositionOfLowestCurrentTopScore	**1**
	31	$VirtualDiceGame	**1**
	32	$AppealDieResult	**1**

9	**37**	`if ($VirtualDiceGame)`
		`{`
		`  $AppealDieResult = rand(1, 5);`
		`}`
		`else`
		`{`
		`  fwrite(STDOUT, "Please roll the appeal die and then enter your result.\n");`
		`  fwrite(STDOUT, "Enter 1 if the result is NOT OUT\n");`
		`  fwrite(STDOUT, "Enter 2 if the result is CAUGHT\n");`
		`  fwrite(STDOUT, "Enter 3 if the result is LBW\n");`
		`  fwrite(STDOUT, "Enter 4 if the result is BOWLED\n");`
		`  fwrite(STDOUT, "Enter 5 if the result is RUN OUT\n");`
		`  $AppealDieResult = intval(trim(fgets(STDIN)));`
		`  fwrite(STDOUT, "\n");`
		`}`
			2

	38	`switch ($AppealDieResult)`
		`{`
		`  case 1: fwrite(STDOUT,"Not out!\n");`
		`  break;`
		`  case 2: fwrite(STDOUT, "Caught!\n");`
		`  break;`
		`  case 3: fwrite(STDOUT, "LBW!\n");`
		`  break;`
		`  case 4: fwrite(STDOUT, "Bowled!\n");`
		`  break;`
		`  case 5: fwrite(STDOUT, "Run out!\n");`
		`  break;`
		`}`
			2

	40	`if ($PlayerOneScore > $PlayerTwoScore)`
		`  fwrite(STDOUT, $PlayerOneName . " wins!\n");`
		`if ($PlayerTwoScore > $PlayerOneScore)`
		`  fwrite(STDOUT, $PlayerTwoName . " wins!\n");`
		`if ($PlayerOneScore == $PlayerTwoScore)`
		`  fwrite(STDOUT, "A draw!\n");`
			3

	42	`fwrite(STDOUT, "Result: ");`
		`$BowlDieResult = intval(trim(fgets(STDIN)));`
		`while ($BowlDieResult < 1 || $BowlDieResult > 6)`
		`{`
		`  fwrite(STDOUT, "Please enter a value between 1 and 6 only\n");`
		`  $BowlDieResult = intval(trim(fgets(STDIN)));`
		`}`
			4

	44	`fwrite(STDOUT, "4. Display top scores\n");`
		`fwrite(STDOUT, "5. Save top scores\n");`
		`fwrite(STDOUT, "9. Quit\n");`
			1

	45	`if (($OptionChosen < 1 \|\| ($OptionChosen > 5) && $OptionChosen != 9)`	**1**

	46	`function SaveTopScores(&$TopScores)`
		`{`
		`  $CurrentFile = fopen("HiScores.txt", "w");`
		`  for ($Count = 1; $Count <= MaxSize; $Count++)`
		`  {`
		`  $LineToAddToFile = "";`
		`  $LineToAddToFile = $TopScores[$Count]["Name"] . ",";`
		`  $LineToAddToFile = $LineToAddToFile . rtrim($TopScores[$Count]["Score"]);`
		`  fwrite($CurrentFile, $LineToAddToFile . "\r\n");`
		`  }`
		`  fclose($CurrentFile);`
		`}`
			10

	47	`} while (!($OptionSelected >= 1 && $OptionSelected <= 5 || $OptionSelected == 9))`
		`fwrite(STDOUT, "\n");`
		`if ($OptionSelected >= 1 && $OptionSelected <= 5)`
		`{`
		`  switch ($OptionSelected)`
		`  {`
		`  case 1: PlayDiceGame($PlayerOneName, $PlayerTwoName, true,  $TopScores);`
		`  break;`
		`  case 2: PlayDiceGame($PlayerOneName, $PlayerTwoName, false,  $TopScores);`
		`  break;`
		`  case 3: LoadTopScores($TopScores);`
		`  break;`
		`  case 4: DisplayTopScores($TopScores);`
		`  break;`
		`  case 5: SaveTopScores($TopScores);`
		`  break;`
		`  }`
		`}`
			4

Python 2 Mark Scheme

Note: Python 2.6 also supports the newer Python 3 `print()` function so allow `print ("...")` as well as `print "..."`
Also accept `int(raw_input("..."))` instead of `input("...")`

7	**20**	`Names = ["", "", "", "", ""]`
		`Names[1] = "Ben"`
		`Names[2] = "Thor"`
		`Names[3] = "Zoe"`
		`Names[4] = "Kate"`
		`# Or:`
		`# Names["", "Ben","Thor", "Zoe","Kate"]`
		
		`# Or:`
		`# Names = [""]`
		`# Names.append("Ben")`
		`# Names.append("Thor")`
		`# Names.append("Zoe")`
		`# Names.append("Kate")`
		
		`Max = 4`
		`Current = 1`
		`Found = False`
		`PlayerName = raw_input("What player are you looking for?")`
		`while (Found == False) and (Current <= Max):`
		`    if Names[Current] == PlayerName:`
		`        Found = True`
		`    else:`
		`        Current += 1`
		`if Found == True: # accept if Found:`
		`    print "Yes, they do have a top score"`
		`else:`
		`    print "No, they do not have a top score"`
		
		A. Answers where `Max` is set to 5 and loop condition of `Current < Max`
		A. Answers where `Max` is set to 4 and loop condition of `Current < Max + 1`
			11

8	**23**	`MAX_SIZE = 4`	**1**
	25	`MAX_SIZE`	**1**

9	**37**	def RollAppealDie(VirtualDiceGame):
		  if VirtualDiceGame:
		  AppealDieResult = random.randint**(1,5)**
		  else:
		  print "Please roll the appeal die and then enter your result."
		  print ""
		  print "Enter 1 if the result is NOT OUT"
		  print "Enter 2 if the result is CAUGHT"
		  print "Enter 3 if the result is LBW"
		  print "Enter 4 if the result is BOWLED"
		  print "Enter 5 if the result is RUN OUT"
		  print ""
		  AppealDieResult = input("Result: ")
		  print ""
		  return AppealDieResult
			2

	38	def DisplayAppealDieResult(AppealDieResult):
		  if AppealDieResult == 1:
		  print "Not out!"
		  elif AppealDieResult == 2:
		  print "Caught!"
		  elif AppealDieResult == 3:
		  print "LBW!"
		  elif AppealDieResult == 4:
		  print "Bowled!"
		  elif AppealDieResult == 5:
		  print "Run out!"
			2

	40	if PlayerOneScore > PlayerTwoScore:
		  print PlayerOneName, "wins!"
		  if PlayerTwoScore > PlayerOneScore:
		  print PlayerTwoName, "wins!"
		  if PlayerOneScore == PlayerTwoScore:
		  print "A draw!"
			3

	42	while BowlDieResult not in [1,2,3,4,5,6]:
		while BowlDieResult not in range(1,7):
		while BowlDieResult < 1 or BowlDieResult >6:
		while not (1 <= BowlDieResult <= 6):
		  BowlDieResult = input("Please enter a value between 1 and 6 only: ")
			4

	44	def DisplayMenu():
		  print "Dice Cricket"
		  print ""
		  print "1. Play game version with virtual dice"
		  print "2. Play game version with real dice"
		  print "3. Load top scores"
		  print "4. Display top scores"
		  print "5. Save top scores"
		  print "9. Quit"
			1

	45	def GetMenuChoice():
		    OptionChosen = input("Please enter your choice: ")
		    if (OptionChosen < 1 or (**OptionChosen > 5** and OptionChosen != 9)):
		        print ""
		        print "That was not one of the allowed options. Please try again: "
		    return OptionChosen
			1

	46	def SaveTopScores(TopScores):
		    OutFile = open("HiScores.txt","w")
		    Count = 1
		    for Count in range(1, MAX_SIZE+1):
		        LineToAddToFile = TopScores[Count].Name + "," + str(TopScores[Count].Score) + "\n":
		        OutFile.write(LineToAddToFile)
		    OutFile.close()
		
		# or more likely
		def SaveTopScores(TopScores):
		  Outfile = open("HiScores.txt","w")
		  For score in (TopScores[1], TopScores[2], TopScores[3], TopScores[4]):
		        Line = score.Name + ","+ str(score.Score) + "\n"
		Outfile.write(line)
		        Outfile.close()
			10

	47	while OptionSelected != 9:
		    DisplayMenu()
		    OptionSelected = GetMenuChoice()
		    while OptionSelected not in [1,2,3,4,**5**,9]:
		        DisplayMenu()
		        OptionSelected = GetMenuChoice()
		    print ""
		    if OptionSelected in [1,2,3,4,**5**]:
		        if OptionSelected == 1:
		            PlayDiceGame(PlayerOneName, PlayerTwoName, True, TopScores)
		        elif OptionSelected == 2:
		            PlayDiceGame(PlayerOneName, PlayerTwoName, False, TopScores)
		        elif OptionSelected == 3:
		            LoadTopScores(TopScores)
		        elif OptionSelected == 4:
		            DisplayTopScores(TopScores)
		        elif OptionSelected == 5:
		            SaveTopScores(TopScores)
			4

Python 3 Mark Scheme

7	**20**	`Names = ["", "", "", "", ""]`
		`Names[1] = "Ben"`
		`Names[2] = "Thor"`
		`Names[3] = "Zoe"`
		`Names[4] = "Kate"`
		`# Or:`
		`# Names["", "Ben","Thor", "Zoe","Kate"]`
		
		`# Or:`
		`# Names = [""]`
		`# Names.append("Ben")`
		`# Names.append("Thor")`
		`# Names.append("Zoe")`
		`# Names.append("Kate")`
		
		`Max = 4`
		`Current = 1`
		`Found = False`
		`PlayerName = input("What player are you looking for?")`
		`while (Found == False) and (Current <= Max):`
		`  if Names[Current] == PlayerName:`
		`  Found = True`
		`  else:`
		`  Current += 1`
		`if Found == True: # accept if Found:`
		`  print("Yes, they do have a top score")`
		`else:`
		`  print("No, they do not have a top score")`
		
		A. Answers where `Max` is set to 5 and loop condition of `Current < Max`
		A. Answers where `Max` is set to 4 and loop condition of `Current < Max + 1`
			11

8	**23**	`MAX_SIZE = 4`	**1**
	25	`MAX_SIZE;`	**1**

9	**37**	`def RollAppealDie(VirtualDiceGame):`
		`  if VirtualDiceGame:`
		`  AppealDieResult = random.randint(1,5)`
		`  else:`
		`  print("Please roll the appeal die and then enter your result.")`
		`  print()`
		`  print("Enter 1 if the result is NOT OUT")`
		`  print("Enter 2 if the result is CAUGHT")`
		`  print("Enter 3 if the result is LBW")`
		`  print("Enter 4 if the result is BOWLED")`
		`  **print("Enter 5 if the result is RUN OUT")**`
		`  print()`
		`  AppealDieResult = int(input("Result: "))`
		`  print()`
		`  return AppealDieResult`
			2

	38	`def DisplayAppealDieResult(AppealDieResult):`
		`  if AppealDieResult == 1:`
		`  print("Not out!")`
		`  elif AppealDieResult == 2:`
		`  print("Caught!")`
		`  elif AppealDieResult == 3:`
		`  print("LBW!")`
		`  elif AppealDieResult == 4:`
		`  print("Bowled!")`
		`  elif AppealDieResult == 5:`
		`  print("Run out!")`
			2

	40	`  if PlayerOneScore > PlayerTwoScore:`
		`  print(PlayerOneName, "wins!")`
		`  if PlayerTwoScore > PlayerOneScore:`
		`  print(PlayerTwoName, "wins!")`
		`  if PlayerOneScore == PlayerTwoScore:`
		`  print("A draw!")`
			3

	42	`while BowlDieResult not in [1,2,3,4,5,6]:`
		`while BowlDieResult not in range(1,7):`
		`while BowlDieResult < 1 or BowlDieResult >6:`
		`while not (1 <= BowlDieResult <= 6):`
		`  BowlDieResult = int(input("Please enter a value between 1 and 6 only: "))`
			4

	44	`  print("4. Display top scores")`
		`  print("5. Save top scores")`
		`  print("9. Quit")`
			1

	45	`def GetMenuChoice():`
		`  OptionChosen = int(input("Please enter your choice: "))`
		`  if (OptionChosen < 1 or (OptionChosen > 5 and OptionChosen != 9)):`
		`  print()`
		`  print("That was not one of the allowed options. Please try again: ")`
		`  return OptionChosen`
			1

	46	`def SaveTopScores(TopScores):`
		`  CurrentFile = open("HiScores.txt","w")`
		`  Count = 1`
		`  for Count in range(1, MAX_SIZE+1):`
		`  LineToAddToFile = TopScores[Count].Name + "," + str(TopScores[Count].Score) + "\n"`
		`  CurrentFile.write(LineToAddToFile)`
		`  CurrentFile.close()`
			10

	47	`while OptionSelected != 9:`
		`  DisplayMenu()`
		`  OptionSelected = GetMenuChoice()`
		`  while OptionSelected not in [1,2,3,4,`**`5`**`,9]:`
		`  DisplayMenu()`
		`  OptionSelected = GetMenuChoice()`
		`  print()`
		`  if OptionSelected in [1,2,3,4,`**`5`**`]:`
		`  if OptionSelected == 1:`
		`  PlayDiceGame(PlayerOneName, PlayerTwoName, True, TopScores)`
		`  elif OptionSelected == 2:`
		`  PlayDiceGame(PlayerOneName, PlayerTwoName, False, TopScores)`
		`  elif OptionSelected == 3:`
		`  LoadTopScores(TopScores)`
		`  elif OptionSelected == 4:`
		`  DisplayTopScores(TopScores)`
		`elif OptionSelected == 5:`
		`SaveTopScores(TopScores)`
			4

UMS conversion calculator www.aqa.org.uk/umsconversion