

1(a)(i)	1 mark each • Class <code>EventItem</code> header (and end where appropriate) • 3 private attributes with suitable data types • Constructor header (and end where appropriate) with 3 (min) parameters within class declaration ... • ... assigning parameters to attributes within constructor e.g. Python <pre>class EventItem(): def __init__(self, pName, pType, pDifficulty): self.__EventName = pName #String self.__EventType = pType #String self.__Difficulty = pDifficulty #Integer</pre> VB.NET <pre>Class EventItem Private EventName As String Private EventType As String Private Difficulty As Integer Sub New(pName, pType, pDifficulty) EventName = pName EventType = pType Difficulty = pDifficulty End Sub End Class</pre>
1(a)(i)	Java <pre>class EventItem{ private String EventName; private String EventType; private Integer Difficulty; public EventItem(String pName, String pType, Integer pDifficulty){ EventName= pName; EventType = pType; Difficulty = pDifficulty; } }</pre>
1(a)(ii)	1 mark each • 1 get method with no parameter ... • ... return correct attribute (without changing) • Remaining 2 correct get methods returning the attributes e.g. Python <pre>def GetName(self): return self.__EventName def GetEventType(self): return self.__EventType def GetDifficulty(self): return self.__Difficulty</pre>

1(a)(ii)	VB.NET <pre>Function GetName() Return EventName End Function Function GetEventType() Return EventType End Function Function GetDifficulty() Return Difficulty End Function</pre> Java <pre>public String GetName(){ return EventName; } public String GetEventType(){ return EventType; } public Integer GetDifficulty(){ return Difficulty; }</pre>
1(b)(i)	1 mark each: 1D array name <code>Group</code> with (min 5 elements and of type <code>EventItem</code>) e.g. Python <pre>Group = [] #type Event, 5 spaces</pre> VB.NET <pre>Dim Group(4) As EventItem</pre> Java <pre>EventItem[] Group = new EventItem[5];</pre>

1(b)(ii)	1 mark each - Any 1 instance of <code>EventItem</code> declared with values passed in correct order stored in the array <code>Group</code> remaining 4 correctly instantiated and stored in <code>Group</code> e.g. Python <pre>Group.append(EventItem("Bridge", "jump", 3)) Group.append(EventItem("Water wade", "swim", 4)) Group.append(EventItem("100 mile run", "run", 5)) Group.append(EventItem("Gridlock", "drive", 2)) Group.append(EventItem("Wall on wall", "jump", 4))</pre> VB.NET <pre>Group(0) = New EventItem("Bridge", "jump", 3) Group(1) = New EventItem("Water wade", "swim", 4) Group(2) = New EventItem("100 mile run", "run", 5) Group(3) = New EventItem("Gridlock", "drive", 2) Group(4) = New EventItem("Wall on wall", "jump", 4)</pre> Java <pre>Group[0] = new EventItem("Bridge", "jump", 3); Group[1] = new EventItem("Water wade", "swim", 4); Group[2] = new EventItem("100 mile run", "run", 5); Group[3] = new EventItem("Gridlock", "drive", 2); Group[4] = new EventItem("Wall on wall", "jump", 4);</pre>
1(c)	1 mark each <code>Class Character</code> declared (and end where appropriate) 5 private attributes with correct data types - Constructor header (and end) taking (min) 5 parameters and parameters assigned to attributes - Get method (with no parameter) returning name attribute

1(c)	e.g. Python <pre>class Character(): def __init__(self, pName, pJump, pSwim, pRun, pDrive): self.__CName = pName #string self.__Jump = pJump #integer chance of success self.__Swim = pSwim #integer chance of success self.__Run = pRun #integer chance of success self.__Drive = pDrive #integer chance of success def GetName(self): return self.__CName #STRING</pre> VB.NET <pre>Class Character Private CName As String Private Jump As Integer Private Swim As Integer Private Run As Integer Private Drive As Integer Sub New(pName, pJump, pSwim, pRun, pDrive) CName = pName Jump = pJump Swim = pSwim Run = pRun Drive = pDrive End Sub Function GetName() Return CName End Function End Class</pre>
------	---

1(c)	Java <pre> class Character{ private String CName; private Integer Jump; private Integer Swim; private Integer Run; private Integer Drive; public Character(String pName, Integer pJump, Integer pSwim, Integer pRun, Integer pDrive){ CName= pName; Jump = pJump; Swim = pSwim; Run = pRun; Drive = pDrive; } public String GetName(){ return CName } } </pre>
1(d)	1 mark each to max 4: • Method header CalculateScore (and end where appropriate) taking (min) 2 parameters • Selection on the type of event using the parameter • ... if skill value is > = difficulty return 100 • ...otherwise subtracting skill value from the difficulty and return correct value 80 (diff 1), 60 (diff 2), 40 (diff 3) and 20 (diff 4) • Using the correct attributes and parameters throughout

1(d)

e.g.

Python

```
def CalculateScore(self, Type, Difficulty):  
    if Type == "jump":  
        Chance = self.__Jump  
    elif Type == "swim":  
        Chance = self.__Swim  
    elif Type == "run":  
        Chance = self.__Run  
    else:  
        Chance = self.__Drive  
    if Chance >= Difficulty:  
        return 100  
    else:  
        Difference = Difficulty - Chance  
        if Difference == 1:  
            return 80  
        elif Difference == 2:  
            return 60  
        elif Difference == 3:  
            return 40  
        elif Difference == 4:  
            return 20  
        else:  
            return 0
```

1(d)

VB.NET

```
Function CalculateScore(Type, Difficulty)
    Dim Chance As Integer
    Dim Difference As Integer
    If Type = "jump" Then
        Chance = Jump
    ElseIf Type = "swim" Then
        Chance = Swim
    ElseIf Type = "run" Then
        Chance = Run
    Else
        Chance = Drive
    End If
    If Chance >= Difficulty Then
        Return 100
    Else
        Difference = Difficulty - Chance
        If Difference = 1 Then
            Return 80
        ElseIf Difference = 2 Then
            Return 60
        ElseIf Difference = 3 Then
            Return 40
        ElseIf Difference = 4 Then
            Return 20
        Else
            Return 0
        End If
    End If
End Function
```

1(d)	Java <pre>public Integer CalculateScore(String Type, Integer Difficulty){ Integer Chance = 0; Integer Difference = 0; if(Type.equals("jump")){ Chance = Jump; }else if(Type.equals("swim")){ Chance = Swim; }else if(Type.equals("run")){ Chance = Run; }else{ Chance = Drive; } if(Chance >= Difficulty){ return 100; }else{ Difference = Difficulty - Chance; if(Difference == 1){ return 80; }else if(Difference == 2){ return 60; }else if(Difference == 3){ return 40; }else if(Difference == 4){ return 20; } } }</pre>
------	--

1(e)(i)	1 mark each • Creating one new instance of <code>Character</code> with correct name and values for 1 character and storing • 2nd correct instance of <code>Character</code> and storing e.g. Python <pre>P1 = Character("Tarz", 5, 3, 5, 1) P2 = Character("Geni", 2, 2, 3, 4)</pre> VB.NET <pre>Dim P1 As Character = New Character("Tarz", 5, 3, 5, 1) Dim P2 As Character = New Character("Geni", 2, 2, 3, 4)</pre> Java <pre>Character P1 = new Character("Tarz", 5, 3, 5, 1); Character P2 = new Character("Geni", 2, 2, 3, 4);</pre>
1(e)(ii)	1 mark each • Looping through each event in <code>Group</code> (or checking each of the 5 events manually) • Using <code>CalculateScore()</code> for each <code>Character</code> object with parameters of type and difficulty • ...comparing the return values from the two function calls ... • ...incrementing points for winning player and outputting their name and message stating they have won for each event. • ...outputting message if it's a draw. • Comparing the total points for each character after all events checked and outputting name of player with most points (and their points) and outputting message if it's a draw • Using get methods correctly throughout

1(e)(ii)	e.g. Python <pre>P1Points = 0 P2Points = 0 for x in range(0, 5): P1EventScore = P1.CalculateScore(Group[x].GetEventType(), Group[x].GetDifficulty()) P2EventScore = P2.CalculateScore(Group[x].GetEventType(), Group[x].GetDifficulty()) if P1EventScore > P2EventScore: P1Points = P1Points + 1 print(P1.GetName(), "you win this event") elif P2EventScore > P1EventScore: P2Points = P2Points + 1 print(P2.GetName(), "you win this event") else: print("This event is a draw") if P1Points > P2Points: print(P1.GetName(), "you have won with", P1Points) elif P2Points > P1Points: print(P2.GetName(), "you have won with", P2Points) else: print("It's a draw")</pre>
----------	---

1(e)(ii)	VB.NET <pre> Dim P1 As Character = New Character("Tarz", 5, 3, 5, 1) Dim P2 As Character = New Character("Geni", 2, 2, 3, 4) Dim P1Points As Integer = 0 Dim P2Points As Integer = 0 Dim P1EventScore As Integer = 0 Dim P2EventScore As Integer = 0 For x = 0 To 4 P1EventScore = P1.CalculateScore(Group(x).GetEventType(), Group(x).GetDifficulty()) P2EventScore = P2.CalculateScore(Group(x).GetEventType(), Group(x).GetDifficulty()) If P1EventScore > P2EventScore Then P1Points = P1Points + 1 Console.WriteLine(P1.GetName() & " you win this event") ElseIf P2EventScore > P1EventScore Then P2Points = P2Points + 1 Console.WriteLine(P2.GetName() & " you win this event") Else Console.WriteLine("This event is a draw") End If Next x If P1Points > P2Points Then Console.WriteLine(P1.GetName() & " you have won with " & P1Points) ElseIf P2Points > P1Points Then Console.WriteLine(P2.GetName() & " you have won with " & P2Points) Else Console.WriteLine("It's a draw") End If </pre>
----------	--

1(e)(ii)	Java <pre> Integer P1Points = 0; Integer P2Points = 0; Integer P1EventScore = 0; Integer P2EventScore = 0; for(Integer x = 0; x < 5; x++){ P1EventScore = P1.CalculateScore(Group[x].GetEventType(), Group[x].GetDifficulty()); P2EventScore = P2.CalculateScore(Group[x].GetEventType(), Group[x].GetDifficulty()); System.out.println("P1 " + P1EventScore + " P2 " + P2EventScore); if(P1EventScore > P2EventScore){ P1Points++; System.out.println(P1.GetName() + " you win this event"); }else if(P2EventScore > P1EventScore){ P2Points++; System.out.println(P2.GetName() + " you win this event"); }else{ System.out.println("This event is a draw"); } } if(P1Points > P2Points){ System.out.println(P1.GetName() + " you have won with " + P1Points); }else if(P2Points > P1Points){ System.out.println(P2.GetName() + " you have won with " + P2Points); }else{ System.out.println("It's a draw"); } </pre>
----------	---

1(e)(iii)	1 mark for output showing the correct winner for each event, the final winner's name (and their points) e.g. <pre> Tarz you win this event Tarz you win this event Tarz you win this event Geni you win this event Tarz you win this event Tarz you have won with 4 </pre>
-----------	---

6(a)	<pre> FUNCTION IsRA(x1, y1, x2, y2, x3, y3 : INTEGER) RETURNS BOOLEAN DECLARE Len1, Len2, Len3 : INTEGER Len1 ← (x1 - x2) ^ 2 + (y1 - y2) ^ 2 Len2 ← (x1 - x3) ^ 2 + (y1 - y3) ^ 2 Len3 ← (x2 - x3) ^ 2 + (y2 - y3) ^ 2 IF (Len1 = Len2 + Len3) OR__ (Len2 = Len1 + Len3) OR__ (Len3 = Len1 + Len2) THEN RETURN TRUE ELSE RETURN FALSE ENDIF ENDFUNCTION </pre> Mark as follows: 1 Calculate the square of the length of at least one side 2 Calculation of all three lengths squared correctly 3 One correct comparison of square of three lengths 4 All three comparisons... 5 combined using logical operators / nested IF // completely correct selection 6 Return result correctly in both cases
6(b)	1 mark for statement of problem: Problem: • The function will return an incorrect value // the test will fail 1 mark for solution: Solution: • Round the calculated values (to a known number of decimal places) • Define a threshold below which any difference can be ignored