

9(a)	1 mark per bullet point, max 2 marks - NOT (A XOR B) - NAND ((B AND C) OR A) $X = (\text{NOT } (A \text{ XOR } B)) \text{ NAND } ((B \text{ AND } C) \text{ OR } A)$ $// X = ((B \text{ AND } C) \text{ OR } A) \text{ NAND } (\text{NOT } (A \text{ XOR } B))$
9(b)	1 mark for 1 or 2 row numbers correct, 2 marks all three correct Row 2 Row 4 Row 7

11(a)	One mark per mark point (Max 6) MP1 LDM #100 seen MP2 Correct use of STO with labelled address (constant or answer) MP3 Correct use of LDD 632 MP4 Correct use of SUB with labelled address (constant) <table border="1"> <thead> <tr> <th>Opcode</th><th>Operand</th></tr> </thead> <tbody> <tr> <td>LDM</td><td>#100</td></tr> <tr> <td>STO</td><td>Constant</td></tr> <tr> <td>LDD</td><td>632</td></tr> <tr> <td>SUB</td><td>Constant</td></tr> <tr> <td>STO</td><td>Answer</td></tr> </tbody> </table> MP5 Storing 100 at a labelled address away from the code MP6 Labelling both addresses away from the code. <table border="1"> <thead> <tr> <th>Label</th><th>Contents</th></tr> </thead> <tbody> <tr> <td>Constant:</td><td>100</td></tr> <tr> <td>Answer:</td><td></td></tr> </tbody> </table>	Opcode	Operand	LDM	#100	STO	Constant	LDD	632	SUB	Constant	STO	Answer	Label	Contents	Constant:	100	Answer:	
Opcode	Operand																		
LDM	#100																		
STO	Constant																		
LDD	632																		
SUB	Constant																		
STO	Answer																		
Label	Contents																		
Constant:	100																		
Answer:																			
11(b)	-55																		

1(a)(i)	1 mark each - Class EventItem 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 Java 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 Group with (min 5 elements and of type EventItem) 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 EventItem declared with values passed in correct order stored in the array Group remaining 4 correctly instantiated and stored in Group 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 - Class Character 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 callsincrementing 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 <code>get</code> methods correctly throughout

1(e)(ii)

e.g.

Python

```

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")

```

1(e)(ii)

VB.NET

```

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

```

1(e)(ii)	<pre> Java 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>
-----------	---

3(a)(i)	1 mark for each bullet point (max 2). - Courses must be available to anyone who wishes to follow them - Courses must be available on the internet - Company is willing to share infrastructure with other companies (public) ...which is more economic for the company
3(a)(ii)	1 mark for each bullet point (max 2 for each disadvantage). - There could be a possible loss of control unlike the LAN ...because the data is stored on a remote infrastructure / someone else's infrastructure ...reliance on external agency to complete tasks, e.g. backups, security - Requires reliable internet connection ...to ensure access to the remote data, more likely with LAN - Increased recurring costs ...as cloud provider charges must be paid, costs for LAN once only.
3(a)(iii)	1 mark each for firewall, encryption and passwords. Firewall: - Monitors incoming and outgoing traffic and rejects any traffic that does not meet the set rules Encryption: - Ensures that if data is intercepted / obtained it cannot be understood without the decryption key Passwords: - Ensures only users with the correct password can access the resources // prevents unauthorised access
3(b)(i)	1 mark for both 1:M relationships as follows: <pre> graph TD CS[COURSE_SCHEDULE] --> T[TUTOR] CS --> CI[COURSE_INFORMATION] </pre>

3(b)(ii)	1 mark for each bullet point. - SELECT Count (CourseID) - AS NumOfCourses - FROM COURSE_SCHEDULE - WHERE DateStarted > "09/09/23"; SELECT Count (CourseID) AS NumOfCourses FROM COURSE_SCHEDULE WHERE DateStarted > "09/09/23";
3(c)	1 mark for each bullet point. - The administrator completes a visual check / checks by eye ...that the tutor identifier input matches the tutor identifier on the original document - Double entry check // the administrator (or a second person) enters the number a second time ...and the system compares it with the first entry

3(a)	1 mark each to max 4 Examples: Installs device drivers ... to allow communication between peripherals and computer - Sends data and receives data to and from peripherals ... such as to an output device and from an input device/by example - Handles buffers for transfer of data ... to ensure smooth transfer between devices that transmit and receive at different speeds - Manages interrupts / signals from the device
3(b)	1 mark each to max 2 - Memory management - File management - Security management - Process management - Error checking and recovery

3(c)	1 mark each to max 3 - Rearranges blocks of individual files (on the HDD) so they are contiguous // moves the free space together - Accessing each file is faster ...because there is no need to search for the next fragment / block of the file ...so less head movement is needed
3(d)(i)	1 mark from - Kibibyte is 1024 bytes and kilobyte is 1000 bytes - Kibibyte is binary prefix and kilobyte is denary prefix
3(d)(ii)	1001 0110 0100
3(d)(iii)	F2
3(d)(iv)	Smallest: 10000000 Largest: 01111111
3(d)(v)	1 mark for working 1 mark for answer 1 0 1 1 0 0 0 0 + 0 0 0 1 1 0 1 1 <u>1 1 0 0 1 0 1 1</u> 1 1
3(d)(vi)	00011001

5(a)(i)	1 mark each to max 2 - Programmer can test sections of the code without every part working / being written - Programmer can debug in real time ... so that errors can be fixed and the program continued from that point - The effect of any changes made by the programmer can be seen immediately - To avoid dependent errors
5(a)(ii)	1 mark each to max 3 - The compiler produces an executable file ... so the user cannot access / edit / sell the code ... and users do not need the translator to run the game - The game can be compiled for different hardware specifications ... and then used to generate more income for the programmer - The program can be tested multiple times without having to retranslate each time

5(b)	1 mark for appropriate licence; 1 mark for each point to max 3 • Commercial software licence • User has to pay for the product so the programmer can gain an income • Enables the program to be copyrighted • ... so the user cannot legally edit the program // the programmer retains control over product • ... and can take legal action against people who attempt to illegally copy it /sell it on • Shareware licence • Enables the program to be copyrighted • The user cannot legally edit the program so the developer retains control over product • User can try the program for free and then pay for the full game which allows the programmer to gain an income • so more people can experience it and therefore be more likely to buy it
------	--