CAMBRIDGE
International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/42**

Paper 4 Practical **October/November 2024**

MARK SCHEME

Maximum Mark: 75

Published

This mark scheme is published as an aid to teachers and candidates, to indicate the requirements of the examination. It shows the basis on which Examiners were instructed to award marks. It does not indicate the details of the discussions that took place at an Examiners' meeting before marking began, which would have considered the acceptability of alternative answers.

Mark schemes should be read in conjunction with the question paper and the Principal Examiner Report for Teachers.

Cambridge International will not enter into discussions about these mark schemes.

Cambridge International is publishing the mark schemes for the October/November 2024 series for most Cambridge IGCSE, Cambridge International A and AS Level components, and some Cambridge O Level components.

This document consists of **39** printed pages.

[Turn over

Generic Marking Principles

These general marking principles must be applied by all examiners when marking candidate answers. They should be applied alongside the specific content of the mark scheme or generic level descriptions for a question. Each question paper and mark scheme will also comply with these marking principles.

GENERIC MARKING PRINCIPLE 1:

Marks must be awarded in line with:

- the specific content of the mark scheme or the generic level descriptors for the question
- the specific skills defined in the mark scheme or in the generic level descriptors for the question
- the standard of response required by a candidate as exemplified by the standardisation scripts.

GENERIC MARKING PRINCIPLE 2:

Marks awarded are always **whole marks** (not half marks, or other fractions).

GENERIC MARKING PRINCIPLE 3:

Marks must be awarded **positively**:

- marks are awarded for correct/valid answers, as defined in the mark scheme. However, credit is given for valid answers which go beyond the scope of the syllabus and mark scheme, referring to your Team Leader as appropriate
- marks are awarded when candidates clearly demonstrate what they know and can do
- marks are not deducted for errors
- marks are not deducted for omissions
- answers should only be judged on the quality of spelling, punctuation and grammar when these features are specifically assessed by the question as indicated by the mark scheme. The meaning, however, should be unambiguous.

GENERIC MARKING PRINCIPLE 4:

Rules must be applied consistently, e.g. in situations where candidates have not followed instructions or in the application of generic level descriptors.

GENERIC MARKING PRINCIPLE 5:

Marks should be awarded using the full range of marks defined in the mark scheme for the question (however; the use of the full mark range may be limited according to the quality of the candidate responses seen).

GENERIC MARKING PRINCIPLE 6:

Marks awarded are based solely on the requirements as defined in the mark scheme. Marks should not be awarded with grade thresholds or grade descriptors in mind.

Question	Answer	Marks

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
	 ```class EventItem():```
	 ``` def __init__(self, pName, pType, pDifficulty):```
	 ``` self.__EventName = pName #String```
	 ``` self.__EventType = pType #String```
	 ``` self.__Difficulty = pDifficulty #Integer```
	
	 VB.NET
	 ```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```
		4

Question	Answer	Marks

1(a)(i)	Java
	
	``java
	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;
	
	     }
	}
	``	

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
	``python
	     def GetName(self):
	         return self.__EventName
	     def GetEventType(self):
	         return self.__EventType
	     def GetDifficulty(self):
	         return self.__Difficulty
	``
		3

Question	Answer	Marks

1(a)(ii)	VB.NET
	
	``
	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;
	}
	``	

1(b)(i)	1 mark each: • 1D array name `Group` with (min 5 elements and of type `EventItem`) e.g. Python ``` Group = [] #type Event, 5 spaces ``` VB.NET ``` Dim Group(4) As EventItem ``` Java ``` EventItem[] Group = new EventItem[5]; ```	**1**

Question	Answer	Marks

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
	`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))`
	
	VB.NET
	`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)`
	
	Java
	`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);`
		3

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	4

Question	Answer	Marks

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>	

Question	Answer	Marks

1(c)	Java
	
	``java
	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
	        }
	}
	``	

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	**4**

Question	Answer	Marks

1(d)	e.g.
	Python
	<pre>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</pre>	

Question	Answer	Marks

1(d)	VB.NET
	
	``vbnet
	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
	``	

Question	Answer	Marks
1(d)	Java	

```java
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;
            }
        }
}
```

Question	Answer	Marks

1(e)(i)	1 mark each
	• Creating one new instance of `Character` with correct name and values for 1 character and storing
	• 2nd correct instance of `Character` and storing
	
	e.g.
	Python
	`P1 = Character("Tarz", 5, 3, 5, 1)`
	`P2 = Character("Geni", 2, 2, 3, 4)`
	
	VB.NET
	`Dim P1 As Character = New Character("Tarz", 5, 3, 5, 1)`
	`Dim P2 As Character = New Character("Geni", 2, 2, 3, 4)`
	
	Java
	`Character P1 = new Character("Tarz", 5, 3, 5, 1);`
	`Character P2 = new Character("Geni", 2, 2, 3, 4);`
		2

1(e)(ii)	1 mark each • Looping through each event in `Group` (or checking each of the 5 events manually) • Using `CalculateScore()` for each `Character` 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	**7**

Question	Answer	Marks

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>	

Question	Answer	Marks

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
	``	

Question	Answer	Marks

1(e)(ii)	Java
	
	``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");
	}
	``	

Question	Answer	Marks

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>
		1

Question	Answer	Marks
2(a)	1 mark each to max • Record structure or class with constructor `Queue` (and end where appropriate) … • … containing a 1D array of (100) integers `QueueArray` • … containing `HeadPointer` and `TailPointer` as integers	3

	e.g.
	Python
	<pre>class Queue:
	    def __init__(self):
	        self.QueueArray = []
	        HeadPointer = 0 #integer
	        TailPointer = 0 #integer
	        for x in range(0, 100):
	            self.QueueArray.append(-1)</pre>
	VB.NET
	<pre>Structure Queue
	    Dim QueueArray() As Integer
	    Dim HeadPointer As Integer
	    Dim TailPointer As Integer
	End Structure</pre>
	Java
	<pre>class queue{
	    private static Integer[] QueueArray = new Integer[100];
	    private static Integer HeadPointer;
	    private static Integer TailPointer;
	
	    public queue(){
	    }
	}</pre>	

Question	Answer	Marks
2(b)	1 mark each • New `Queue` record/object created/instance of class • Queue **field/attribute** head pointer initialised to -1, tail pointer to 0 • All 100 array **field/attribute** elements initialised with -1	**3**
| | e.g.<br>Python<br><br>```python
class Queue:
    def __init__(self):
        self.QueueArray = []
        for x in range(0, 100):
            self.QueueArray.append(-1)

        self.HeadPointer = -1
        self.TailPointer = 0
TheQueue= Queue()
```<br><br>VB.NET<br>```vbnet
 Dim TheQueue As New Queue
        TheQueue.HeadPointer = -1
        TheQueue.TailPointer = 0
        ReDim TheQueue.QueueArray(100)
        For x = 0 To 99
            TheQueue.QueueArray(x) = -1
        Next
```<br><br>Java<br>```java
class queue{
     private static Integer[] QueueArray = new Integer[100];
     private static Integer HeadPointer;
     private static Integer TailPointer;
``` | |

Question	Answer	Marks
| 2(b) | ```java
        public queue(){
                HeadPointer = -1;
                TailPointer = 0;
                for(Integer x = 0; x < 100; x++){
                        QueueArray[x] = -1;
                }
        }
}
public static void main(String args[]){
        queue TheQueue = new queue();
}
``` | |
| 2(c) | 1 mark for each completed statement to max 3<br>1 mark for correct values returned in correct places<br>1 mark for function header taking (at least) one parameter and the rest of function correct and using the record/class data structure accurately.<br><br>Pseudocode<br>```
FUNCTION Enqueue(BYREF AQueue : Queue, BYVAL TheData : INTEGER)
                    RETURNS INTEGER
    IF AQueue.HeadPointer = -1 THEN
        AQueue.QueueArray[AQueue.TailPointer] ← TheData
        AQueue.HeadPointer ← 0
        AQueue.TailPointer ← AQueue.TailPointer + 1
        RETURN 1
    ELSE
        IF AQueue.TailPointer > 99 THEN
          RETURN -1
        ELSE
          AQueue.QueueArray[AQueue.TailPointer] ← TheData
          AQueue.TailPointer ← AQueue.TailPointer + 1
          RETURN 1
        ENDIF
    ENDIF
ENDFUNCTION
``` | **5** |

Question	Answer	Marks

2(c)	e.g.
	Python
	<pre>def Enqueue(AQueue, TheData):
	
	    if AQueue.HeadPointer == -1:
	        AQueue.HeadPointer = 0
	        AQueue.QueueArray[AQueue.HeadPointer] = TheData
	        AQueue.TailPointer +=1
	        return AQueue, 1
	    elif AQueue.TailPointer > 99:
	        return AQueue, -1
	    else:
	        AQueue.QueueArray[AQueue.TailPointer] = TheData
	        AQueue.TailPointer = AQueue.TailPointer + 1
	        return AQueue, 1</pre>
	VB.NET
	<pre>Function Enqueue(ByRef AQueue As Queue, ByVal TheData As Integer)
	    If AQueue.HeadPointer = -1 Then
	        AQueue.QueueArray(AQueue.TailPointer) = TheData
	        AQueue.HeadPointer = 0
	         AQueue.TailPointer += 1
	        Return 1
	    ElseIf AQueue.TailPointer > 99 Then
	        Return -1
	    Else
	        AQueue.QueueArray(AQueue.TailPointer) = TheData
	        AQueue.TailPointer += 1
	        Return 1
	    End If
	
	End Function</pre>	

Question	Answer	Marks

2(c)	Java
	<pre>public static Integer Enqueue(Integer TheData){
	      if(GetHeadPointer() == -1){
	            SetData(TheData);
	            SetHeadPointer(0);
	            SetTailPointer(GetTailPointer() + 1);
	            return 1;
	      }else if(GetTailPointer() > 99){
	            return -1;
	      }else{
	            SetData(TheData);
	            SetTailPointer(GetTailPointer() + 1);
	            return 1;
	      }
	}</pre>	

2(d)	1 mark each to max 3
	• Function header (and end) iterating through each element in the queue
	• Starting at `HeadPointer` and incrementing until `TailPointer – 1` …
	• … concatenating and returning all integer values with a space between
	
	e.g.
	Python
	
	<pre>def ReturnAllData(TheQueue):
	    Temp = ""
	
	    for X in range(TheQueue.HeadPointer, TheQueue.TailPointer):
	        Temp = Temp + str(TheQueue.QueueArray[X]) + " "
	    return Temp</pre>
		3

Question	Answer	Marks

2(d)	**VB.NET**
	
	``vbnet
	Function ReturnAllData(AQueue As Queue)
	      Dim Temp As String = ""
	      For X = AQueue.HeadPointer To AQueue.TailPointer - 1
	          Temp = Temp & AQueue.QueueArray(X).ToString() & " "
	      Next X
	      Return Temp
	  End Function
	``
	
	Java
	``java
	public static String ReturnAllData(){
	      String Temp = "";
	      Integer Counter = 0;
	       for(int X = HeadPointer; X < TailPointer; X++){
	       Temp = Temp + Integer.toString(QueueArray[X]) + " ";
	      }
	      return Temp;
	}
	``	

2(e)(i)	1 mark each • Taking only **10** inputs in loop/one at a time • Calling `Enqueue()` with each input (min) **and** storing/using return value … • … only calling `Enqueue()` **once** when each input is an integer $>= 0$. Do **not** award if this validation stops 10 valid inputs being enqueued. • Outputting message if each item is inserted **and** outputting a message if queue is full • Calling `ReturnAllData()` and outputting return value at the end	**5**

Question	Answer	Marks

2(e)(i)	e.g.
	Python
	<pre>for x in range(0, 10):
	    Continue = True
	    while(Continue == True):
	        DataInput = int(input("Enter an integer that is 0 or more"))
	        if DataInput > -1:
	            Continue = False
	
	    TheQueue, ReturnValue = Enqueue(TheQueue, DataInput)
	
	    if(ReturnValue == -1):
	        print("Queue full")
	    else:
	        print("Item inserted")
	
	print(ReturnAllData(TheQueue))</pre>
	VB.NET
	<pre>Dim ContinueLoop As Boolean
	Dim DataInput As Integer
	Dim ReturnValue As Integer
	For x = 0 To 9
	    ContinueLoop = True
	    While ContinueLoop = True
	        Console.WriteLine("Enter an integer that is 0 or more")
	        DataInput = Console.ReadLine
	        If DataInput > -1 Then
	            ContinueLoop = False</pre>	

Question	Answer	Marks

2(e)(i)	<pre> End If
	    End While
	    ReturnValue = Enqueue(TheQueue, DataInput)
	    If ReturnValue = 2 Then
	        Console.WriteLine("Queue full")
	    Else
	        Console.WriteLine("Item inserted")
	    End If
	Next
	
	Console.WriteLine(ReturnAllData(TheQueue))
	
	Java
	Boolean Continue = true;
	Integer DataInput = -1;
	Scanner scanner = new Scanner(System.in);
	Integer ReturnValue;
	
	for(Integer x = 0; x < 10; x++){
	    Continue = true;
	    while(Continue == true){
	        System.out.println("Enter an integer that is 0 or more");
	        DataInput = Integer.parseInt(scanner.nextLine());
	        if(DataInput > -1){
	            Continue = false;
	        }
	    }
	    ReturnValue = Enqueue(DataInput);
	    if(ReturnValue == -1){
	        System.out.println("Queue full");
	    }else{
	        System.out.println("Item inserted");
	    }
	}
	System.out.println(ReturnAllData());</pre>	

Question	Answer	Marks

2(e)(ii)	1 mark for each
	• All values input and 10 messages 'Inserted' (i.e. –1 is not inserted)
	• Screenshot show 10 9 8 7 6 5 4 3 2 1 on one line with a space between each number
	
	e.g.
	`Enter an integer that is 0 or more10`
	`Item inserted`
	`Enter an integer that is 0 or more9`
	`Item inserted`
	`Enter an integer that is 0 or more-1`
	`Enter an integer that is 0 or more8`
	`Item inserted`
	`Enter an integer that is 0 or more7`
	`Item inserted`
	`Enter an integer that is 0 or more6`
	`Item inserted`
	`Enter an integer that is 0 or more5`
	`Item inserted`
	`Enter an integer that is 0 or more4`
	`Item inserted`
	`Enter an integer that is 0 or more3`
	`Item inserted`
	`Enter an integer that is 0 or more2`
	`Item inserted`
	`Enter an integer that is 0 or more1`
	`Item inserted`
	`10 9 8 7 6 5 4 3 2 1`
		2

2(f)	1 mark each
	• Function `Dequeue()` (head and close), returning a value in all cases
	• Checking if empty **and** returning –1
	• Returning item at `HeadPointer` **without deleting**/changing it
	• Incrementing `HeadPointer`
	
	Example program code:
	Python
	<pre>def Dequeue(AQueue):
	    if AQueue.HeadPointer = 100 or AQueue.HeadPointer == -1 or AQueue.HeadPointer ==
	AQueue.TailPointer:
	        return AQueue, -1
	    else:
	        Temp = AQueue.QueueArray[AQueue.HeadPointer]
	        AQueue.HeadPointer = AQueue.HeadPointer + 1
	        return AQueue, Temp</pre>
		4

Question	Answer	Marks

2(f)	VB.NET
	
	``
	Function Dequeue(ByRef AQueue As Queue)
	    If AQueue.HeadPointer = 100 or AQueue.HeadPointer = -1 or AQueue.HeadPointer =
	AQueue.TailPointer Then
	        Return -1
	    Else
	        Dim Temp As Integer = AQueue.QueueArray(AQueue.HeadPointer)
	        AQueue.HeadPointer += 1
	        Return Temp
	    End If
	End Function
	``
	
	Java
	
	``
	public static Integer Dequeue(){
	    if(GetHeadPointer() = 100 || GetTailpointer() == -1 || GetHeadPointer() ==
	GetTailpointer()){
	        return -1;
	    }else{
	        Integer Temp = GetData(GetHeadPointer());
	        SetHeadPointer(GetHeadPointer() + 1);
	        return Temp;
	    }
	}
	``	

Question	Answer	Marks

2(g)(i)	1 mark each
	• Calls `Dequeue()` **twice and** stores/uses return value
	• Outputs "Queue empty" when each return values is –1 and outputs return value otherwise
	• Calls `ReturnAllData()` at the end
	
	e.g.
	Python
	``
	TheQueue, ReturnValue = Dequeue(TheQueue)
	if ReturnValue == -1:
	    print("Queue empty")
	else:
	    print(ReturnValue, " is returned")
	TheQueue, ReturnValue = Dequeue(TheQueue)
	if ReturnValue == -1:
	    print("Queue empty")
	else:
	    print(ReturnValue, " is returned")
	print(ReturnAllData(TheQueue))
	``
	
	VB.NET
	``
	ReturnValue = Dequeue(TheQueue)
	If ReturnValue = -1 Then
	    Console.WriteLine("Queue empty")
	Else
	    Console.WriteLine(ReturnValue, " is returned")
	End If
	``
		3

Question	Answer	Marks
| 2(g)(i) | ```
ReturnValue = Dequeue(TheQueue)
If ReturnValue = -1 Then
    Console.WriteLine("Queue empty")
Else
    Console.WriteLine(ReturnValue, " is returned")
End If
Console.WriteLine(ReturnAllData(TheQueue))
```<br><br>Java<br>```
ReturnValue = Dequeue();
if(ReturnValue == -1){
    System.out.println("Queue empty");
}else{
    System.out.println(ReturnValue + " is returned");
}

ReturnValue = Dequeue();
if(ReturnValue == -1){
    System.out.println("Queue empty");
}else{
    System.out.println(ReturnValue + " is returned");
}
ReturnAllData(TheQueue);
``` | |

Question	Answer	Marks

2(g)(ii)	1 mark each
	
	• Screenshot shows the input of 10 9 8 7 6 5 4 3 2 1
	Output for 10 returned
	Output for 9 is returned
	
	e.g.
	<pre>Enter an integer that is 0 or more10
	Item inserted
	Enter an integer that is 0 or more9
	Item inserted
	Enter an integer that is 0 or more-1
	Enter an integer that is 0 or more8
	Item inserted
	Enter an integer that is 0 or more7
	Item inserted
	Enter an integer that is 0 or more6
	Item inserted
	Enter an integer that is 0 or more5
	Item inserted
	Enter an integer that is 0 or more4
	Item inserted
	Enter an integer that is 0 or more3
	Item inserted
	Enter an integer that is 0 or more2
	Item inserted
	Enter an integer that is 0 or more1
	Item inserted
	10 9 8 7 6 5 4 3 2 1
	10 is returned
	9 is returned
	8 7 6 5 4 3 2 1</pre>
		1

Question	Answer	Marks

3(a)	1 mark each
	• HighScores created as 2D array, (of strings) with (min) 7×3 elements (local to main) …
	• … all elements initialised to empty string ("")
	
	e.g.
	Python
	<pre>HighScores = [] #String, 7 x 3
	HighScores = [['' for x in range(3)] for y in range(7)]</pre>
	VB.NET
	<pre>Dim HighScores(7, 3) As String
	For(X = 0 to 7)
	    For(Y = 0 to 3)
	        HighScores(X, Y) = ""
	  Next Y
	Next X</pre>
	Java
	<pre>String[][] HighScores = new String[7][3];
	for(Int X = 0; X <7; X++){
	    for(Int Y = 0; Y < 3; Y++){
	        HighScores[X][Y] = "";
	  }
	}</pre>
		2

3(b)	1 mark each • Function header (and end where appropriate) that returns populated array • Opening text file to read and closing the file in an appropriate place • Looping through 7 players/to EOF/21 times … • … reading in each group of 3 data items and storing each in separate element in 2D array for each player • Exception handling with all file handling within the try, appropriate catch and an output	5

Question	Answer	Marks

3(b)	e.g.
	Python
	<pre>def ReadData():
	    Temp = []
	    HighScores = []
	    try:
	        File = open("HighScoreTable.txt")
	        Temp = File.read().split("\n")
	        File.close()
	    except:
	        print("No file found")
	    NumberRecords = len(Temp)-1
	    Counter = 0
	
	    while Counter < NumberRecords:
	
	        HighScores.append([Temp[Counter], Temp[Counter+1], Temp[Counter+2]])
	        Counter = Counter + 3
	
	    return HighScores</pre>
	VB.NET
	<pre> Function ReadData()
	        Dim TextFile As String = "HighScoreTable.txt"
	
	        Dim HighScores(7, 3) As String
	        Try
	            Dim FileReader As New System.IO.StreamReader(TextFile)
	            Dim Counter As Integer = 0</pre>	

Question	Answer	Marks

3(b)	<pre> While Counter < 8
	                    HighScores(Counter, 0) = FileReader.ReadLine()
	                    HighScores(Counter, 1) = FileReader.ReadLine()
	                    HighScores(Counter, 2) = FileReader.ReadLine()
	
	                    Counter = Counter + 1
	
	            End While
	            FileReader.Close()
	        Catch ex As Exception
	            Console.WriteLine("No file found")
	        End Try
	        Return HighScores
	  End Function</pre>
	Java
	<pre>public static String[][] ReadData(){
	      String TextFile = "HighScoreTable.txt";
	      String[][] HighScores = new String[7][3];
	
	      try{
	            FileReader f = new FileReader(TextFile);
	            BufferedReader Reader = new BufferedReader(f);
	            for(Integer X = 0; X < 7; X++){
	                  try{</pre>	

Question	Answer	Marks

3(b)	```
	                    HighScores[X][0] = Reader.readLine();
	                    HighScores[X][1] = Reader.readLine();
	                    HighScores[X][2] = Reader.readLine();
	                }catch(IOException ex){}
	            }
	            try{
	                    Reader.close();
	            }catch(IOException ex){}
	            return HighScores;
	        }catch(FileNotFoundException e){
	            System.out.println("File not found");
	        }
	        return HighScores;
	}
	``	

3(c)	1 mark each
	• Procedure (header and end) taking (min) 1 parameter (2D array), looping through each of the first dimension in array …
	• … outputting all data in correct format
	
	e.g.
	Python
	
	``python
	def OutputHighScores(HighScores):
	    for x in range(0, len(HighScores)):
	        print(HighScores[x][0], "reached level", HighScores[x][1], "with a score of", HighScores[x][2])
	``
	
	VB.NET
	``
	  Sub OutputHighScores(HighScores(,))
	        For x = 0 To 6
	            Console.WriteLine(HighScores(x, 0) & " reached level " & HighScores(x, 1) & " with a score of " & HighScores(x, 2))
	        Next
	  End Sub
	``
		2

Question	Answer	Marks

3(c)	Java
	``java
	public static void OutputHighScores(String[][] HighScores){
	        for(Integer x = 0; x < 7; x++){
	                System.out.println(HighScores[x][0] + " reached level " + HighScores[x][1] + "
	with a score of " + HighScores[x][2]);
	        }
	}
	``	

3(d)	1 mark each
	• Function header (and end taking array as parameter) returning sorted array.
	• Comparing the levels and swapping **all** dimensions when in incorrect order
	• Comparing scores when levels are the same and swapping **all** dimensions when in incorrect order
	• Correct loops and comparisons to put data in correct order
	
	e.g.
	Python
	``python
	def SortScores(HighScores):
	    Counter = 0
	    ArrayLength = len(HighScores)
	
	    for x in range(ArrayLength-1):
	        for y in range(0, ArrayLength-x-1):
	            if int(HighScores[y][1]) < int(HighScores[y + 1][1]):
	                HighScores[y], HighScores[y + 1] = HighScores[y + 1], HighScores[y]
	            elif int(HighScores[y][1]) == int(HighScores[y+1][1]):
	                if int(HighScores[y][2]) < int(HighScores[y+1][2]):
	                    HighScores[y], HighScores[y + 1] = HighScores[y + 1], HighScores[y]
	    return HighScores
	``
	
	VB.NET
	``vbnet
	  Function SortScores(HighScores)
	        Dim ArrayLength As Integer = 6
	        Dim Temp1 As String
	``
		4

Question	Answer	Marks
3(d)	(see code below)	

```
        Dim Temp2 As String
        Dim Temp3 As String
        For x = 0 To ArrayLength - 1
            For y = 0 To ArrayLength - x - 1
                If Integer.Parse(HighScores(y, 1)) < Integer.Parse(HighScores(y + 1, 1))
Then
                    Temp1 = HighScores(y, 0)
                    Temp2 = HighScores(y, 1)
                    Temp3 = HighScores(y, 2)
                    HighScores(y, 0) = HighScores(y + 1, 0)
                    HighScores(y, 1) = HighScores(y + 1, 1)
                    HighScores(y, 2) = HighScores(y + 1, 2)
                    HighScores(y + 1, 0) = Temp1
                    HighScores(y + 1, 1) = Temp2
                    HighScores(y + 1, 2) = Temp3
                ElseIf Integer.Parse(HighScores(y, 1)) = Integer.Parse(HighScores(y + 1,
1)) Then
                    If Int(HighScores(y, 2)) < Int(HighScores(y + 1, 2)) Then
                        Temp1 = HighScores(y, 0)
                        Temp2 = HighScores(y, 1)
                        Temp3 = HighScores(y, 2)
                        HighScores(y, 0) = HighScores(y + 1, 0)
                        HighScores(y, 1) = HighScores(y + 1, 1)
                        HighScores(y, 2) = HighScores(y + 1, 2)
                        HighScores(y + 1, 0) = Temp1
                        HighScores(y + 1, 1) = Temp2
                        HighScores(y + 1, 2) = Temp3
                    End If
                End If
            Next y
        Next x
        Return HighScores
    End Function

Java
public static String[][] SortScores(String[][] HighScores){
```

Question	Answer	Marks
3(d)	(see code below)	

```
        Integer ArrayLength = 6;
        String Temp1;
        String Temp2;
        String Temp3;

        for(Integer x = 0; x < ArrayLength; x++){
                for(Integer y = 0; y < ArrayLength - x; y++){
                        if(Integer.parseInt(HighScores[y][1]) <
Integer.parseInt(HighScores[y+1][1])){
                                Temp1 = HighScores[y][0];
                                Temp2 = HighScores[y][1];
                                Temp3 = HighScores[y][2];
                                HighScores[y][0] = HighScores[y+1][0];
                                HighScores[y][1] = HighScores[y+1][1];
                                HighScores[y][2] = HighScores[y+1][2];
                                HighScores[y+1][0] = Temp1;
                                HighScores[y+1][1] = Temp2;
                                HighScores[y+1][2] = Temp3;
                        }else if(Integer.parseInt(HighScores[y][1]) ==
Integer.parseInt(HighScores[y+1][1])){
                                if(Integer.parseInt(HighScores[y][2]) <
Integer.parseInt(HighScores[y+1][2])){
                                        Temp1 = HighScores[y][0];
                                        Temp2 = HighScores[y][1];
                                        Temp3 = HighScores[y][2];
                                        HighScores[y][0] = HighScores[y+1][0];
                                        HighScores[y][1] = HighScores[y+1][1];
                                        HighScores[y][2] = HighScores[y+1][2];
                                        HighScores[y+1][0] = Temp1;
                                        HighScores[y+1][1] = Temp2;
                                        HighScores[y+1][2] = Temp3;
                                }
                        }
```

Question	Answer	Marks

3(d)	<pre> }
	        }
	        return HighScores;
	}</pre>	

3(e)(i)	1 mark each
	• Code statements in order:
	<pre>HighScores = ReadData()
	HighScores = SortScores(HighScores)</pre> // `HighScores = SortScores()`
	
	• Code statements in order:
	<pre>OUTPUT "Before"
	OutputHighScores(HighScores)</pre> unsorted
	<pre>OUTPUT "After"
	OutputHighScores(HighScores)</pre> sorted
	
	e.g.
	Python
	<pre>HighScores = []
	HighScores = ReadData()
	print("Before")
	OutputHighScores(HighScores)
	HighScores = SortScores(HighScores)
	print("After")
	OutputHighScores(HighScores)</pre>
	VB.NET
	<pre> Sub Main(args As String())
	        Dim HighScores(7, 3) As String
	        HighScores = ReadData()
	        Console.WriteLine("Before")
	        OutputHighScores(HighScores)
	        HighScores = SortScores(HighScores)
	        Console.WriteLine("After")
	        OutputHighScores(HighScores)
	    End Sub</pre>
		2

Question	Answer	Marks

3(e)(i)	Java
	``java
	public static void main(String args[]){
	        String[][] HighScores = new String[7][3];
	        HighScores = ReadData();
	        System.out.println("Before");
	        OutputHighScores(HighScores);
	        HighScores = SortScores(HighScores);
	        System.out.println("After");
	        OutputHighScores(HighScores);
	}
	``	

3(e)(ii)	Output showing 'Before' and players and scores in correct format before sorting Output showing 'After' players and scores in correct order and format after sorting e.g. ``` Before GHEH got to level 3 with a score of 10 KWQW got to level 4 with a score of 20 MMND got to level 4 with a score of 18 RFOO got to level 5 with a score of 20 XXHD got to level 3 with a score of 19 QWSD got to level 3 with a score of 15 JGHF got to level 5 with a score of 22 After JGHF got to level 5 with a score of 22 RFOO got to level 5 with a score of 20 KWQW got to level 4 with a score of 20 MMND got to level 4 with a score of 18 XXHD got to level 3 with a score of 19 QWSD got to level 3 with a score of 15 GHEH got to level 3 with a score of 10 ```	**2**