



Cambridge International AS & A Level

COMPUTER SCIENCE**9618/43**

Paper 4 Practical

October/November 2023

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 2023 series for most Cambridge IGCSE, Cambridge International A and AS Level components, and some Cambridge O Level components.

This document consists of **37** printed pages.

PUBLISHED**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 descriptors 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.

PUBLISHED**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.

PUBLISHED

Question	Answer	Marks
1(a)(i)	One mark each to max 5 <ul style="list-style-type: none">• Function header (and end where appropriate) taking one string parameter• Calculating length of parameter string• Looping correct number of times• Checking the first character against all vowels• Accessing the remainder of the string• Remainder of function correct with nothing extra i.e. totalling, must match structure of given algorithm	5

Question	Answer	Marks
	<p>Example program code:</p> <p>Java</p> <pre>public static Integer IterativeVowels(String Value){ Integer Total = 0; Integer LengthString = Value.length(); char FirstCharacter; for(Integer X = 0; X < LengthString; X++){ FirstCharacter = Value.charAt(X); if(FirstCharacter == 'a' FirstCharacter == 'e' FirstCharacter == 'i' FirstCharacter == 'o' FirstCharacter == 'u'){ Total++; } Value = Value.substring(1, Value.length()); } return Total; }</pre> <p>VB.NET</p> <pre>Function IterativeVowels(Value) Dim Total As Integer = 0 Dim FirstCharacter As Char For x = 0 To Len(Value) - 1 FirstCharacter = Left(Value, 1) If FirstCharacter = "a" Or FirstCharacter = "e" Or FirstCharacter = "i" Or FirstCharacter = "o" Or FirstCharacter = "u" Then Total = Total + 1 End If Value = Right(Value, Len(Value) - 1) Next Return Total End Function</pre>	

PUBLISHED

Question	Answer	Marks
Python	<pre>def IterativeVowels(Value): Total = 0 for X in range(0, len(Value)): FirstCharacter = Value[0] if FirstCharacter == 'a' or FirstCharacter == 'e' or FirstCharacter == 'i' or FirstCharacter == 'o' or FirstCharacter == 'u': Total = Total + 1 Value = Value[1:len(Value)] return Total</pre>	
1(a)(ii)	<p>One mark each</p> <ul style="list-style-type: none"> • Calling the function with "house" • Outputting the return value <p>Example program code:</p> <p>Java</p> <pre>System.out.println(IterativeVowels("house"));</pre> <p>VB.NET</p> <pre>Console.WriteLine(IterativeVowels("house"))</pre> <p>Python</p> <pre>print(IterativeVowels("house"))</pre>	2
1(a)(iii)	One mark for screenshot outputting 3	1
1(b)(i)	<p>One mark each</p> <ul style="list-style-type: none"> • Recursive call • Function header (and end where appropriate) taking string parameter (returning integer where given) • Base case checking (length is 0) and returning 0 • Extracting first character and checking if a vowel ... • ... if it is a vowel, returning 1 + recursive call with 1 less character • ... if not a vowel, return recursive call with 1 less character 	6

Question	Answer	Marks
	<p>Example program code:</p> <p>Java</p> <pre>public static Integer RecursiveVowels(String Value){ char FirstCharacter; if(Value.length() == 0){ return 0; }else{ FirstCharacter = Value.charAt(0); if(FirstCharacter == 'a' FirstCharacter == 'e' FirstCharacter == 'i' FirstCharacter == 'o' FirstCharacter == 'u'){ return 1 + RecursiveVowels(Value.substring(1, Value.length())); }else{ return RecursiveVowels(Value.substring(1, Value.length())); } } }</pre> <p>VB.NET</p> <pre>Function RecursiveVowels(Value) Dim firstCharacter As Char If Len(Value) = 0 Then Return 0 Else firstCharacter = Left(Value, 1) If firstCharacter = "a" Or firstCharacter = "e" Or firstCharacter = "i" Or firstCharacter = "o" Or firstCharacter = "u" Then Return 1 + RecursiveVowels(Right(Value, Len(Value) - 1)) Else Return RecursiveVowels(Right(Value, Len(Value) - 1)) End If End If End Function</pre>	

PUBLISHED

Question	Answer	Marks
	<pre> End If End If End Function Python def RecursiveVowels(Value): if len(Value) == 0: return 0 else: FirstCharacter = Value[0] if FirstCharacter == 'a' or FirstCharacter == 'e' or FirstCharacter == 'i' or FirstCharacter == 'o' or FirstCharacter == 'u': return 1 + RecursiveVowels(Value[1:len(Value)]) else: return RecursiveVowels(Value[1:len(Value)]) </pre>	
1(b)(ii)	<p>One mark for calling recursive function with "imagine" and outputting return value</p> <p>Example program code:</p> <p>Java System.out.println(RecursiveVowels("imagine"));</p> <p>VB.NET Console.WriteLine(RecursiveVowels("imagine"))</p> <p>Python print(RecursiveVowels("imagine"))</p>	1
1(b)(iii)	One mark for screenshot showing 4	1

PUBLISHED

Question	Answer	Marks
2(a)(i)	<p>One mark each</p> <ul style="list-style-type: none"> • (Global) array with identifier <code>Queue</code> with (minimum) 50 elements (of type string) • <code>TailPointer</code> (integer) initialised to 0, <code>HeadPointer</code> (integer) initialised to -1 <p>Example program code:</p> <p>Java</p> <pre>public static String[] Queue = new String[50]; public static Integer HeadPointer = -1; public static Integer TailPointer = 0;</pre> <p>VB.NET</p> <pre>Dim Queue(50) As String Dim HeadPointer As Integer Dim TailPointer As Integer Sub Main(args As String()) HeadPointer = -1 TailPointer = 0 End Sub</pre> <p>Python</p> <pre>global Queue #string 50 elements global HeadPointer global TailPointer #main Queue = [] HeadPointer = -1 TailPointer = 0</pre>	2

PUBLISHED

Question	Answer	Marks
2(a)(ii)	<p>One mark each</p> <ul style="list-style-type: none"> • Procedure <code>Enqueue()</code> header (and close where appropriate) with one (string) parameter • Checking if queue is full and outputting suitable message • ... otherwise inserting parameter to next space • ... increment <code>TailPointer</code> and set <code>HeadPointer</code> to 0 if first item (<code>HeadPointer = -1</code>) <p>Example program code:</p> <p>Java</p> <pre>public static void Enqueue(String Value){ if(TailPointer == 50){ System.out.println("Queue full"); }else{ Queue[TailPointer] = Value; TailPointer++; if(HeadPointer == -1){ HeadPointer = 0;} } }</pre> <p>VB.NET</p> <pre>Sub Enqueue(Data) If TailPointer = 50 Then Console.WriteLine("Queue full") Else Queue(TailPointer) = Data TailPointer = TailPointer + 1 If (HeadPointer = -1) Then HeadPointer = 0 End If End If End Sub</pre>	4

PUBLISHED

Question	Answer	Marks
2(a)(ii)	<p>Python</p> <pre>def Enqueue(Data): global TailPointer global HeadPointer global Queue if TailPointer == 50: print("Queue full") else: Queue.append(Data) TailPointer +=1 if HeadPointer == -1: HeadPointer = 0</pre>	

PUBLISHED

Question	Answer	Marks
2(a)(iii)	<p>One mark each to max 4</p> <ul style="list-style-type: none"> • Function header <code>Dequeue()</code> (and end where appropriate) with no parameter • Checking if empty ... • ... outputting suitable message and returning "Empty" • (otherwise) incrementing head pointer • returning next value (at head pointer before incrementing) <p>Example program code:</p> <p>Java</p> <pre>public static String Dequeue(){ if(HeadPointer == -1 HeadPointer == TailPointer){ System.out.println("Queue empty"); return "Empty"; }else{ HeadPointer ++; return Queue[HeadPointer - 1];}}</pre> <p>VB.NET</p> <pre>Function Dequeue() If HeadPointer = -1 Or HeadPointer = TailPointer Then Console.WriteLine("Queue empty") Return "Empty" Else HeadPointer = HeadPointer + 1 Return Queue(HeadPointer - 1) End If End Function</pre>	4

PUBLISHED

Question	Answer	Marks
2(a)(iii)	Python <pre>def Dequeue(): global Queue global HeadPointer if HeadPointer == -1 or HeadPointer == TailPointer: print("Queue empty") return "Empty" else: HeadPointer +=1 return Queue[HeadPointer - 1]</pre>	

PUBLISHED

Question	Answer	Marks
2(b)	<p>One mark each to max 6</p> <ul style="list-style-type: none"> • Procedure header <code>ReadData()</code> with no parameters • Opening file ... • ... and closing file • Looping until EOF/set amount • Reading in each value • ... calling <code>Enqueue()</code> with each value <ul style="list-style-type: none"> • Use of exception handling with appropriate output <p>Example program code:</p> <p>Java</p> <pre>public static void ReadData(){ try{ Scanner Scanner1 = new Scanner(new File("QueueData.txt")); while(Scanner1.hasNextLine()){ Enqueue(Scanner1.next()); } Scanner1.close(); }catch(FileNotFoundException ex){ System.out.println("No file found"); } }</pre> <p>VB.NET</p> <pre>Sub ReadData() Try Dim DataReader As New System.IO.StreamReader("QueueData.txt") Do Until DataReader.EndOfStream Enqueue(DataReader.ReadLine()) Loop Catch End Try End Sub</pre>	6

PUBLISHED

Question	Answer	Marks
2(b)	<pre> Loop DataReader.Close() Catch ex As Exception Console.WriteLine("No file") End Try End Sub Python def ReadData(): try: DataFile = open("QueueData.txt") for Line in DataFile: Enqueue(Line.strip()) DataFile.close() except IOError: print("No file") </pre>	

PUBLISHED

Question	Answer	Marks
2(c)(i)	<p>One mark each</p> <ul style="list-style-type: none"> • Declaration of record type/class RecordData • ID as a string and total as an Integer <p>Example program code:</p> <p>Java</p> <pre>class RecordData{ public String ID; public Integer Total; public RecordData(String IDP, Integer TotalP){ ID = IDP; Total = TotalP; } }</pre> <p>VB.NET</p> <pre>Structure RecordData Dim ID As String Dim Total As Integer End Structure</pre> <p>Python</p> <pre>class RecordData: #self._ID string #self._Total integer def __init__(self, IDP, TotalP): self._ID = IDP self._Total = TotalP</pre>	2

PUBLISHED

Question	Answer	Marks
2(c)(i)	<pre>def SetID(self, Value): self._ID = Value def GetID(self): return self._ID def SetTotal(self, Value): self._Total = Value def GetTotal(self): return self._Total</pre>	
2(c)(ii)	<p>One mark each</p> <ul style="list-style-type: none"> (global) 1D Array named <code>Records</code> of type <code>RecordData</code> (global) <code>NumberRecords</code> declared as integer and initialised to 0 <p>Example program code:</p> <p>Java</p> <pre>public static RecordData[] Records = new RecordData[50]; public static Integer NumberRecords = 0;</pre> <p>VB.NET</p> <pre>Dim Records(49) As RecordData Dim NumberRecords As Integer Sub Main(args As String()) NumberRecords = 0 End Sub</pre> <p>Python</p> <pre>#main Records = [] #50 elements of type RecordData NumberRecords = 0</pre>	2

Question	Answer	Marks
2(c)(iii)	<p>One mark each to max 5</p> <ul style="list-style-type: none"> • Incrementing <code>NumberRecords</code> each time (twice) a new record is added • Procedure header (and end) and using <code>Dequeue()</code> and storing/using return value <code>DataAccessed ← Dequeue()</code> • Checking if <code>NumberRecords</code> is 0 and creating a new record with ID and total as 1: <code>IF NumberRecords = 0 THEN</code> <code>Records[NumberRecords].ID ← DataAccessed</code> <code>Records[NumberRecords].Total ← 1</code> <code>Flag ← TRUE</code> • Looping through all array elements to find matching ID and incrementing total if found <code>FOR X ← 0 TO NumberRecords - 1 Check Python loop end</code> <code>IF Records[X].ID = DataAccessed THEN</code> <code>Records[X].Total ← Records[X].Total + 1</code> <code>Flag ← TRUE</code> <code>ENDIF</code> <code>NEXT X</code> • Adding new record if record is not found, storing ID and total as 1 <code>IF Flag = FALSE THEN</code> <code>Records[NumberRecords].ID ← DataAccessed</code> <code>Records[NumberRecords].Total ← 1</code> <code>NumberRecords ← NumberRecords + 1</code> <code>ENDIF</code> 	5

PUBLISHED

Question	Answer	Marks
2(c)(iii)	<ul style="list-style-type: none"> • Example program code: <p>Java</p> <pre>public static void TotalData(){ String DataAccessed = Dequeue(); Boolean Flag = false; if(NumberRecords == 0){ Records[NumberRecords] = new RecordData(DataAccessed, 1); NumberRecords ++; Flag = true; }else{ for(Integer X = 0; X < NumberRecords; X++){ if(Records[X].ID.equals(DataAccessed)){ Records[X].Total++; Flag = true; } } } if(Flag == false){ Records[NumberRecords] = new RecordData(DataAccessed, 1); NumberRecords ++; } }</pre> <p>VB.NET</p> <pre>Sub TotalData() Dim DataAccessed As String Dim Flag As Boolean = False DataAccessed = Dequeue()</pre>	

PUBLISHED

Question	Answer	Marks
2(c)(iii)	<pre> If NumberRecords = 0 Then Records(NumberRecords).ID = DataAccessed Records(NumberRecords).Total = Records(NumberRecords).Total + 1 NumberRecords = NumberRecords + 1 Flag = True Else For X = 0 To NumberRecords - 1 If Records(X).ID = DataAccessed Then Records(X).Total = Records(X).Total + 1 Flag = True End If Next End If If Flag = False Then Records(NumberRecords).ID = DataAccessed Records(NumberRecords).Total = Records(NumberRecords).Total + 1 NumberRecords = NumberRecords + 1 End If End Sub Python def TotalData(): global NumberRecords global Records Flag = False DataAccessed = Dequeue() if NumberRecords == 0: Records.append(RecordData(DataAccessed, 1)) </pre>	

PUBLISHED

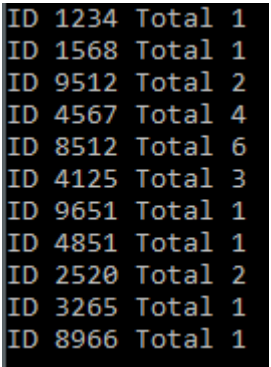
Question	Answer	Marks
2(c)(iii)	<pre> NumberRecords += 1 Flag = True else: for X in range(0, NumberRecords): if(Records[X].GetID() == DataAccessed): Records[X].SetTotal(Records[X].GetTotal() + 1) Flag = True if Flag == False: Records.append(RecordData(DataAccessed, 1)) NumberRecords += 1 </pre>	
2(d)	<p>One mark each</p> <ul style="list-style-type: none"> Looping through all array elements and outputting ID and total in correct format 	1

Question	Answer	Marks
	<p>Example program code:</p> <p>Java</p> <pre>public static void OutputRecords(){ for(Integer X = 0; X < NumberRecords; X++){ System.out.println("ID ", Records[X].ID + " Total " + Records[X].Total); } }</pre> <p>VB.NET</p> <pre>Sub OutputRecords() For X = 0 To NumberRecords - 1 Console.WriteLine("ID " & Records(X).ID & " Total " & Records(X).Total) Next End Sub</pre> <p>Python</p> <pre>def OutputRecords(): for X in range(0, NumberRecords): print("ID", Records[X].GetID(), " Total ", Records[X].GetTotal())</pre>	

PUBLISHED

Question	Answer	Marks
2(e)(i)	<p>One mark each</p> <ul style="list-style-type: none"> • Calling ReadData() first and OutputRecords() last • Looping through all queue elements and calling TotalData() for each queue element <p>Example program code:</p> <p>Java</p> <pre>public static void main(String args[]){ ReadData(); while(HeadPointer != TailPointer){ TotalData(); } OutputRecords(); }</pre> <p>VB.NET</p> <pre>Sub Main(args As String()) HeadPointer = 0 TailPointer = 0 ReadData() NumberRecords = 0 While HeadPointer <> TailPointer TotalData() End While OutputRecords() End Sub</pre>	2

PUBLISHED

Question	Answer	Marks
2(e)(i)	<pre>Python #main Queue = [] Records = [] HeadPointer = 0 TailPointer = 0 ReadData() NumberRecords = 0 while HeadPointer != TailPointer: TotalData() OutputRecords()</pre>	
2(e)(ii)	<p>One mark for screenshot e.g.</p>  <pre>ID 1234 Total 1 ID 1568 Total 1 ID 9512 Total 2 ID 4567 Total 4 ID 8512 Total 6 ID 4125 Total 3 ID 9651 Total 1 ID 4851 Total 1 ID 2520 Total 2 ID 3265 Total 1 ID 8966 Total 1</pre>	1

PUBLISHED

Question	Answer	Marks
3(a)(i)	<p>One mark each to max 4</p> <ul style="list-style-type: none"> • Class header (and end where appropriate) • Three attributes with correct names and data types • Constructor header (and end where appropriate) with 3 parameters • Within constructor, assigns attributes to parameters <p>Example program code:</p> <p>Java</p> <pre>class Character{ private Integer XPosition; private Integer YPosition; private String Name; public Character(Integer XPositionP, Integer YPositionP, String NameP){ XPosition = XPositionP; YPosition = YPositionP; Name = NameP; } }</pre> <p>VB.NET</p> <pre>Class Character Private XPosition As Integer Private YPosition As Integer Private Name As String Sub New(XPositionP, YPositionP, NameP) XPosition = XPositionP YPosition = YPositionP Name = NameP End Sub End Class</pre>	4

PUBLISHED

Question	Answer	Marks
3(a)(i)	<p>Python</p> <pre>class Character: #self.XPosition integer #self.YPosition integer #self.Name string def __init__(self, XPositionP, YPositionP, NameP): self.XPosition = XPositionP self.YPosition = YPositionP self.Name = NameP</pre>	

PUBLISHED

Question	Answer	Marks
3(a)(ii)	<p>One mark each</p> <ul style="list-style-type: none"> • 1 get header with no parameter ... • ... returning correct value • 2nd get method <p>Example program code:</p> <p>Java</p> <pre>public Integer GetXPosition(){ return XPosition; } public Integer GetYPosition(){ return YPosition; }</pre> <p>VB.NET</p> <pre>Function GetXPosition() Return XPosition End Function Function GetYPosition() Return YPosition End Function</pre> <p>Python</p> <pre>def GetXPosition(self): return self._XPosition def GetYPosition(self): return self._YPosition</pre>	3

PUBLISHED

Question	Answer	Marks
3(a)(iii)	<p>One mark each to max 4</p> <ul style="list-style-type: none"> • 1 set method header (and end where appropriate) with parameter ... • ... adding parameter to X/Y Position attribute and storing in the X/Y attribute • If (resulting value is) more than 10 000 limiting to 10 000 and if less than 0 limiting to 0 • Second correct set method <p>Example program code:</p> <p>Java</p> <pre>public void SetXPosition(Integer Value){ XPosition = XPosition + Value; if(XPosition > 10000){ XPosition = 10000; }else if(XPosition < 0){ XPosition = 0; } } public void SetYPosition(Integer Value){ YPosition = YPosition + Value; if(YPosition > 10000){ YPosition = 10000; }else if(YPosition < 0){ YPosition = 0; } }</pre> <p>VB.NET</p> <pre>Function SetXPosition(Value) XPosition = XPosition + Value If XPosition > 10000 Then XPosition = 10000</pre>	4

PUBLISHED

Question	Answer	Marks
3(a)(iii)	<pre> ElseIf XPosition < 0 Then XPosition = 0 End If End Function Function SetYPosition(Value) YPosition = YPosition + Value If YPosition > 10000 Then YPosition = 10000 ElseIf YPosition < 0 Then YPosition = 0 End If End Function </pre> <p>Python</p> <pre> def SetXPosition(self, Value): self._XPosition = self._XPosition + Value if(self.XPosition > 10000): self.XPosition = 10000 elif self.XPosition < 0: self.XPosition = 0 def SetYPosition(self, Value): self.YPosition = self.YPosition + Value if(self.YPosition > 10000): self.YPosition = 10000 elif self.YPosition < 0: self.YPosition = 0 </pre>	

PUBLISHED

Question	Answer	Marks
3(a)(iv)	<p>One mark each</p> <ul style="list-style-type: none"> • Method header with (string) parameter • Checking parameter for direction ... • ... using SetYPosition() and SetXPosition() correctly ... • ... with correct parameters <p>Example program code:</p> <p>Java</p> <pre>public void Move(String Direction){ if(Direction.equals("up")){ SetYPosition(10); }else if(Direction.equals("down")){ SetYPosition(-10); }else if(Direction.equals("right")){ SetXPosition(10); }else{ SetXPosition(-10); } }</pre> <p>VB.NET</p> <pre>Overridable Sub Move(Direction) If Direction = "up" Then SetYPosition(10) ElseIf Direction = "down" Then SetYPosition(-10) ElseIf Direction = "right" Then SetXPosition(10) ElseIf Direction = "left" Then SetXPosition(-10) End If End Sub</pre>	4

PUBLISHED

Question	Answer	Marks
3(a)(iv)	<p>Python</p> <pre>def Move(self, Direction): if(Direction == "up"): self.SetYPosition(10) elif(Direction == "down"): self.SetYPosition(-10) elif(Direction == "right"): self.SetXPosition(10) else: self.SetXPosition(-10)</pre>	
3(b)	<p>One mark each</p> <ul style="list-style-type: none"> • New instance of <code>Character</code> created with identifier <code>Jack</code> ... • ... correct constructor called and values passed <p>Example program code:</p> <p>Java</p> <pre>Character Jack = new Character(50, 50, "Jack");</pre> <p>VB.NET</p> <pre>Dim Jack As Character = New Character(50, 50, "Jack")</pre> <p>Python</p> <pre>Jack = Character(50, 50, "Jack")</pre>	2

PUBLISHED

Question	Answer	Marks
3(c)(i)	<p>One mark each</p> <ul style="list-style-type: none"> • Class header inheriting from Character • Constructor taking all 3 parameters ... • ... calling parent/super constructor with the 3 parameters <p>Example program code:</p> <p>Java</p> <pre>class BikeCharacter extends Character{ public BikeCharacter(Integer XPositionP, Integer YPositionP, String NameP){ super(XPositionP, YPositionP, NameP); } }</pre> <p>VB.NET</p> <pre>Class BikeCharacter Inherits Character Sub New(XPositionP, YPositionP, NameP) MyBase.New(XPositionP, YPositionP, NameP) End Sub End Class</pre> <p>Python</p> <pre>class BikeCharacter(Character): def __init__(self, XPositionP, YPositionP, NameP): super().__init__(XPositionP, YPositionP, NameP)</pre>	3

PUBLISHED

Question	Answer	Marks
3(c)(ii)	<p>One mark each</p> <ul style="list-style-type: none"> • Method header taking parameter and overriding parent/super Move () • Correct changes to method to update values by 20 <p>Example program code:</p> <p>Java</p> <pre>public void Move(String Direction){ if(Direction.equals("up")){ super.SetYPosition(20); }else if(Direction.equals("down")){ super.SetYPosition(-20); }else if(Direction.equals("right")){ super.SetXPosition(20); }else{ super.SetXPosition(-20); } }</pre> <p>VB.NET</p> <pre>Overrides Sub Move(Direction) If Direction = "up" Then SetYPosition(20) ElseIf Direction = "down" Then SetYPosition(-20) ElseIf Direction = "right" Then SetXPosition(20) ElseIf Direction = "left" Then SetXPosition(-20) End If End Sub</pre>	2

PUBLISHED

Question	Answer	Marks
3(c)(ii)	<p>Python</p> <pre>def Move(self, Direction): if(Direction == "up"): super().SetYPosition(20) elif(Direction == "down"): super().SetYPosition(-20) elif(Direction == "right"): super().SetXPosition(2) else: super().SetXPosition(-20)</pre>	
3(d)	<p>One mark each</p> <ul style="list-style-type: none"> • Declaring new <code>BikeCharacter</code> with correct values e.g. <p>Java</p> <pre>BikeCharacter Karla = new BikeCharacter(100, 50, "Karla");</pre> <p>VB.NET</p> <pre>Dim Karla As BikeCharacter = New BikeCharacter(100, 50, "Karla")</pre> <p>Python</p> <pre>Karla = BikeCharacter(100, 50, "Karla")</pre>	1

PUBLISHED

Question	Answer	Marks
3(e)(i)	<p>One mark each to max 5</p> <ul style="list-style-type: none"> • Reading in both values (character and direction) with appropriate prompts • Character name is validated as e.g. Jack/Karla, and direction is validated as e.g. up/down/left/right • Calling <code>Move()</code> for the character input, with direction input as a parameter • Outputting character's new X and Y position in a suitable format ... • ... using get methods <p>Example program code:</p> <p>Java</p> <pre> System.out.println("Would you like to move Jack or Karla?"); CharacterToMove = (scanner.nextLine()).toLowerCase(); while(CharacterToMove.equals("jack") == false && CharacterToMove.equals("karla") == false){ System.out.println("Invalid, try again"); CharacterToMove = (scanner.nextLine()).toLowerCase(); } System.out.println("Which direction? Up, down, left or right?"); Direction = (scanner.nextLine()).toLowerCase(); while(Direction.equals("up") == false && Direction.equals("down") == false && Direction.equals("left") == false && Direction.equals("right")== false){ System.out.println("Invalid, try again"); Direction = (scanner.nextLine()).toLowerCase(); } if(CharacterToMove.equals("jack")){ Jack.Move(Direction); System.out.println("Jack's new position is X = " + Jack.GetXPosition() + " Y = " + Jack.GetYPosition()); }else{ Karla.Move(Direction); System.out.println("Karla's new position is " + Karla.GetXPosition() + " " + Karla.GetYPosition()); } </pre>	5

PUBLISHED

Question	Answer	Marks
3(e)(i)	<pre> VB.NET Console.WriteLine("Would you like to move Jack or Karla?") CharacterToMove = Console.ReadLine.ToLower() While CharacterToMove <> "jack" And CharacterToMove <> "karla" Console.WriteLine("Invalid try again") CharacterToMove = Console.ReadLine End While Console.WriteLine("Which direction? Up, down, left or right") Direction = Console.ReadLine.ToLower() While Direction <> "up" And Direction <> "down" And Direction <> "left" And Direction <> "right" Console.WriteLine("Invalid try again") Direction = Console.ReadLine End While If CharacterToMove = "jack" Then Jack.Move(Direction) Console.WriteLine("Jack's new position is X = " & Jack.GetXPosition & " Y = " & Jack.GetYPosition) Else Karla.Move(Direction) Console.WriteLine("Karla's new position is X = " & Karla.GetXPosition & " Y = " & Karla.GetYPosition) End If Console.WriteLine("Would you like to Continue? Enter True to continue, or anything else to quit") </pre>	

PUBLISHED

Question	Answer	Marks
3(e)(i)	<p>Python</p> <pre> CharacterToMove = input("Would you like to move Jack or Karla?").lower() while CharacterToMove != "jack" and CharacterToMove != "karla": CharacterToMove = input("Invalid try again") Direction = input("Which direction? Up, down, left or right?") while Direction != "up" and Direction != "down" and Direction != "left" and Direction != "right": Direction = input("Invalid try again") if CharacterToMove == "jack": Jack.Move(Direction) print("Jack's new position is X =", Jack.GetXPosition(), "Y =", Jack.GetYPosition()) else: Karla.Move(Direction) print("Karla's new position is X =", Karla.GetXPosition(), "Y =", Karla.GetYPosition()) </pre>	
3(e)(ii)	<p>One mark for each test</p> <pre> Would you like to move Jack or Karla? jack Which direction? Up, down, left or right right Jack's new position is X = 60 Y = 50 Would you like to move Jack or Karla? karla Which direction? Up, down, left or right down Karla's new position is X = 100 Y = 30 </pre>	2