Cambridge Assessment International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/04**

Paper 4 Practical **For examination from 2021**

MARK SCHEME

Maximum Mark: 75

Specimen

This document has **28** pages. Blank pages are indicated.

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.

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)	1 mark per bullet to max 2
	• Creating array with identifier `TheData` (as numeric data type) as local
	• Allocating the correct data to the array
	
	e.g. VB.NET
	`Sub Main`
	`  Dim TheData(0 to 8) As Integer`
	`  TheData(0) = 20`
	`  TheData(1) = 3`
	`  TheData(2) = 4`
	`  TheData(3) = 8`
	`  TheData(4) = 12`
	`  TheData(5) = 99`
	`  TheData(6) = 4`
	`  TheData(7) = 26`
	`  TheData(8) = 4`
	`End Sub`
	
	e.g. Java
	`public static void main(String[] args){`
	`  int[] TheData = new int[]`
	`  TheData[0] = 20;`
	`  TheData[1] = 3;`
	`  TheData[2] = 4;`
	`  TheData[3] = 8;`
	`  TheData[4] = 12;`
	`  TheData[5] = 99;`
	`  TheData[6] = 4;`
	`  TheData[7] = 26;`
	`  TheData[8] = 4;`
	
	e.g. Python
	`TheData = [20, 3, 4, 8, 12, 99, 4, 26, 4]`
		2

Question	Answer	Marks

1(b)	1 mark per bullet to max 7
	• Procedure insertion sort
	• Takes array as a parameter
	• For loop with missing `length-1` (or equivalent)
	• While loop with correct missing variable 'inserted'
	• Correct `IF` statement following structure
	• Swapping elements correct, replacing missing element with '`DataToInsert`'
	• Following all remaining elements of pseudocode
	
	e.g. VB.NET
	`Sub InsertionSort(ByRef TheData() As Integer)`
	`  Dim NextV As Integer`
	`  For Count = 0 To TheData.Length - 1`
	`  Dim DataToInsert As Integer = TheData(Count)`
	`  Dim Inserted As Integer = 0`
	`  NextValue = Count - 1`
	`  While (NextValue >= 0 And Inserted <> 1)`
	`  If (DataToInsert < TheData(NextValue)) Then`
	`  TheData(NextValue + 1) = TheData(NextValue)`
	`  NextValue = NextValue - 1`
	`  TheData(NextValue + 1) = DataToInsert`
	`  Else`
	`  Inserted = 1`
	`  End If`
	`  End While`
	`  Next`
	`End Sub`
		7

Question	Answer	Marks

1(b)	e.g. Java
	`public static void InsertionSort(int[] TheData){`
	`  int nextValue;`
	`  for (int Count = 0; Count < TheData.length; count++){`
	`  int DataToInsert = TheData[Count];`
	`  int Inserted = 0;`
	`  NextValue = Count - 1;`
	`  while(NextValue >= 0 && Inserted != 1){`
	`  if(DataToInsert < TheData[NextValue]){`
	`  TheData[NextValue + 1] = TheData[NextValue];`
	`  NextValue = NextValue - 1;`
	`  TheData[NextValue + 1] = DataToInsert;`
	`  }else{`
	`  Inserted = 1;`
	`  }`
	`  }`
	`  }`
	`}`
	
	e.g. Python
	`def InsertionSort(TheData):`
	`  for Count in range(0, len(TheData)):`
	`  DataToInsert = TheData[Count]`
	`  Inserted = 0`
	`  NextValue = Count - 1`
	`  while(NextValue >= 0 and Inserted != 1):`
	`  if(DataToInsert < TheData[NextValue]):`
	`  TheData[NextValue+1] = TheData[NextValue]`
	`  NextValue = NextValue - 1`
	`  TheData[NextValue+1] = DataToInsert`
	`  else:`
	`  Inserted = 1`	

Question	Answer	Marks

1(c)	1 mark per bullet
	• Suitable Procedure declared with array parameter
	• Loop through each element
	• Outputs the array element
	
	e.g. VB.NET
	`Sub PrintArray(TheData() As Integer)`
	`  For Count = 0 To 9`
	`  Console.WriteLine(TheData(Count))`
	`  Next`
	`End Sub`
	
	e.g. Java
	`public static void PrintArray(int[] TheData){`
	`  for(int Count = 0;Count < TheData.Length; Count++){`
	`  System.output.println(TheData[CCount]);`
	`  {`
	`}`
	
	e.g. Python
	`def PrintArray(TheData):`
	`  for count in range(0, len(TheData)):`
	`  print(TheData[Count])`
		3

Question	Answer	Marks

1(d)(i)	1 mark per bullet to max 3
	• Calling `PrintArray` (equiv) before and after sort
	• Calling `InsertionSort` from **part 1(c)**
	• Outputting appropriate messages for both print calls
	
	e.g. VB.NET
	`Console.WriteLine("Before")`
	`PrintArray()`
	`InsertionSort()`
	`Console.WriteLine("After")`
	`PrintArray()`
	
	e.g. Java
	`System.out.println("Before");`
	`PrintArray(TheData);`
	`InsertionSort(theData);`
	`System.out.println("After");`
	`PrintArray(TheData);`
	
	e.g. Python
	`print("Before")`
	`PrintArray(TheData)`
	`InsertionSort(TheData)`
	`print("After")`
	`PrintArray(TheData)`
		3

1(d)(ii)	1 mark per bullet • Output of array before sorting (with a suitable heading) • Output of array after sorting (with a suitable heading)	**2**

Question	Answer	Marks

1(e)(i)	1 mark per bullet to max 6
	• Function declaration taking `TheData` as a parameter (by value)
	• Taking a number as input
	• … validating/casting the input (as a whole number)
	• Looping through each array element (or other appropriate method)
	• Comparison of array value against input
	• Returning true when found **and** outputting 'found'
	• … efficiently (i.e. not continuing to search when it is found)
	• Returning false when not found **and** outputting 'not found'
	
	e.g. VB.NET
	`Function Search(ByVal TheData() As Integer) as Boolean`
	`  Console.WriteLine("Enter a whole number")`
	`  Dim NumberInput As Integer`
	`  NumberInput = Console.ReadLine()`
	
	`  For Count = 0 To 9`
	`  If TheData(Count) = NumberInput Then`
	`  Console.WriteLine("Found")`
	`  Return True`
	`  End If`
	`  Next`
	`  Console.WriteLine("Not found")`
	`  Return False`
	`End Function`
		6

Question	Answer	Marks

1(e)(i)	e.g. Java
	`public static void Search(int[] TheData){`
	`  Scanner in = new Scanner(System.in);`
	`  int NumberInput;`
	`  System.out.println("Enter a whole number");`
	`  NumberInput = in.nextInt();`
	`  for (int Count = 0; Count < TheData.length; count++){`
	`  if (TheData[Count] == NumberInput){`
	`  System.out.prinln("Found");`
	`  return true;`
	`  }`
	`  }`
	`  System.out.println("Not found");`
	`  return false;`
	`  }`
	e.g. Python
	`def Search(TheData):`
	`  NumberInput = int(input("Enter a whole number"))`
	`  for count in range(0, len(TheData)):`
	`  if(TheData[Count] == NumberInput):`
	`  print("Found")`
	`  return True`
	`  print("Not found")`
	`  return False`	

1(e)(ii)	1 mark per bullet • 8 outputting found • 9 outputting not found	**2**

Question	Answer	Marks

2(a)	1 mark per bullet
	• Class header and end (all code must be within the class)
	• Declaring `BoxName`, Creator and `GameLocation` as string
	• Declaring `DateHidden` as Date (or equivalent) and `Active` as a Boolean
	• Declaring `LastFinds` as a 2D array with 10x2 elements as string
	
	e.g. VB.NET
	`Public Class HiddenBox`
	`  Private BoxName As String`
	`  Private Creator As String`
	`  Private DateHidden As Date`
	`  Private GameLocation As String`
	`  Private LastFinds(0 To 9, 0 To 1) As String`
	`  Private Active As Boolean`
	`End Class`
	
	e.g. Java
	`public static class HiddenBox {`
	`  private String BoxName;`
	`  private String Creator;`
	`  private String DateHidden;`
	`  private String GameLocation;`
	`  private String[][] LastFinds = new String[10][2];`
	`  private Boolean Active;`
	`}`
	
	e.g. Python
	`class HiddenBox:`
	`  # __BoxName String`
	`  # __Creator String`
	`  # __DateHidden String`
	`  # __GameLocation String`
	`  # __LastFinds [10][2] String`
	`  # __Active String`
		4

Question	Answer	Marks

2(b)	1 mark per bullet
	• Constructor declaration taking **four** parameters
	• Setting `BoxName`, `Creator`, `DateHidden` and `GameLocation` to parameter values
	• Set `Active` to false
	• Initialising all `LastFinds` elements to empty/null/equivalent
	
	e.g. VB.NET
	`Public Sub New(NewBoxName, NewCreator, NewDateHidden, NewLocation)`
	`    BoxName = NewBoxName`
	`    Creator = NewCreator`
	`    DateHidden = NewDateHidden`
	`    GameLocation = NewLocation`
	`    Active = False`
	
	`    For x = 0 To 9`
	`        For y = 0 To 2`
	`            LastFinds(x, y) = "null"`
	`        Next y`
	`    Next x`
	`End Sub`
	
	e.g. Java
	`public HiddenBox(String NewBoxName, NewCreator, NewDateHidden, NewLocation){`
	`    this.BoxName = NewBoxName`
	`    this.Creator = NewCreator;`
	`    this.DateHidden = NewDateHidden;`
	`    this.Location = NewLocation;`
	`    this.Active = false;`
	`    for(int x = 0; x < 10; x++){`
	`        for(int y = 0; y < 2; y++){`
	`            this.LastFinds[x][y] = "";`
	`        }`
	`    }`
	`}`
		4

Question	Answer	Marks

2(b)	e.g. Python
	`def __init__(self, NewBoxName, NewCreator, NewDateHidden, NewLocation):`
	`  self.__BoxName = NewBoxName`
	`  self.__Creator = NewCreator`
	`  self.__DateHidden = NewDateHidden`
	`  self.__GameLocation = NewLocation`
	`  self.__LastFinds = [["" for j in range(0, 2)] for I in range(0, 10)]`
	`  self.__Active = False`	

2(c)	1 mark per bullet
	• `GetBoxName` declared as a function in the class `HiddenBox` (or as a property)
	• … returns `BoxName`
	• `GetGameLocation` declared as a function in the class `HiddenBox` (or as a property) returns `GridReference`
	
	e.g. VB.NET
	`Public Function GetBoxName() As String`
	`  Return BoxName`
	`End Function`
	
	`Public Function GetGameLocation() As String`
	`  Return GameLocation`
	`End Function`
	
	e.g. Java
	`public String getBoxName(){`
	`  return this.BoxName`
	`}`
	`public String getGameLocation(){`
	`  return this.Location`
	`}`
	
	e.g. Python
	`def GetBoxName():`
	`  return self.__BoxName`
	`def GetLocation():`
	`  return self.__GameLocation`
		3

Question	Answer	Marks

2(d)(i)	1 mark per bullet to max 2
	• Declaring TheBoxes with 10 000 spaces
	• … as type HiddenBox and as local variable
	
	e.g. VB.NET
	`Sub Main()`
	`  Dim TheBoxes(0 To 9999) As HiddenBox`
	
	`End Sub`
	
	e.g. Java
	`public static void main(String[] args){`
	`  HiddenBox[] TheBoxes = new HiddenBox[10000];`
	`}`
	
	e.g. Python
	`TheBoxes = [HiddenBox("","","","") for I in range(0, 10000)]`
		2

Question	Answer	Marks

2(d)(ii)	1 mark per bullet to max 4
	• Declaring New and taking `TheBoxes` (and box counter) as parameters
	• Reading in the Box Name, Creator, Date Hidden and Game Location
	• Creating a new instance of `HiddenBox` using the constructor
	• … sending the correct parameters in the correct order
	• Appending the instance to `TheBoxes`
	• Incrementing `NumBoxes`
	
	e.g. VB.NET
	`Public Sub NewBox(ByRef TheBoxes() As HiddenBox, ByRef NumBoxes As Integer)`
	`  Console.WriteLine("Enter the name of the box")`
	`  Dim BoxName As String = Console.ReadLine()`
	`  Console.WriteLine("Enter the creator's name")`
	`  Dim Creator As String = Console.ReadLine()`
	`  Console.WriteLine("Enter the date the box was hidden")`
	`  Dim DateHidden As Date = Console.ReadLine()`
	`  Console.WriteLine("Enter the location of the box")`
	`  Dim GameLocation As String = Console.ReadLine()`
	
	`  TheBoxes(NumBoxes) = New HiddenBox(BoxName, Creator, DateHidden, GameLocation)`
	`  NumBoxes = NumBoxes + 1;`
	`End Sub`
		4

Question	Answer	Marks

2(d)(ii)	e.g. Java
	`public static int NewBox(HiddenBox[] TheBoxes, int NumBoxes){`
	`  Scanner scanner = new Scanner(System.in);`
	`  System.out.println("Enter the name of the box");`
	`  String BoxName = scanner.nextLine();`
	`  System.out.println("Enter the creator's name");`
	`  String Creator = scanner.nextLine();`
	`  System.out.println("Enter the location of the box");`
	`  String GameLocation = scanner.nextLine();`
	`System.out.println("Enter the date the box was hidden");`
	`  String DateHidden = scanner.nextLine();`
	`  TheBoxes[NumBoxes] = new HiddenBox(BoxName, Creator, DateHidden, GameLocation);`
	`  return(NumBoxes + 1);`
	`}`
	
	e.g. Python
	`def NewBox(TheBoxes, NumBoxes):`
	`  BoxName = input("Enter the name of the box")`
	`  Creator = input("Enter the creator's name")`
	`  DateHidden = input("Enter the date the box was hidden")`
	`  GameLocation = input("Enter the location of the box")`
	`  TheBoxes[NumBoxes] = HiddenBox(BoxName, Creator, DateHidden, GameLocation)`
	`  return(NumBoxes + 1)`	

2(d)(iii)	1 mark
	• Calling NewBox
	
	e.g. VB.NET
	`  NewBox(TheBoxes, NumBoxes)`
	
	e.g. Java
	`  NumBoxes = NewBox(TheBoxes, NumBoxes);`
	
	e.g. Python
	`NumBoxes = NewBox(TheBoxes, NumBoxes)`
		1

Question	Answer	Marks

2(e)	1 mark per bullet to max 3
	• Class declaration with inheritance from `HiddenBox`
	• Declaration of new properties (with appropriate data types)
	• Overriding the constructor to also take new parameters
	• Editing `HiddenBox` to allow for inheritance of properties/methods
	
	e.g. VB.NET
	`Public Class PuzzleBox`
	`  Inherits HiddenBox`
	`  Private PuzzleText As String`
	`  Private Solution As String`
	
	`  Public Sub New(NewBoxName, NewCreator, NewDateHidden, NewGridReference, NewPuzzleText, NewSolution)`
	`  MyBase.New(NewBoxName, NewCreator, NewDateHidden, NewGridReference)`
	`  PuzzleText = NewPuzzleText`
	`  Solution = NewSolution`
	`  End Sub`
	`End Class`
	
	e.g. Java
	`public static class PuzzleBox extends HiddenBox{`
	`  private String PuzzleText;`
	`  private String Solution;`
	`  public Puzzle(String NewBoxName, String NewCreator, String NewDataHidden, String NewGameLocation, String NewPuzzleText, String NewSolution){`
	`  super(NewBoxName, NewCreator, NewDateHidden, NewGameLocation);`
	`  this.PuzzleText = NewPuzzleText;`
	`  this.Solution = NewSolution;`
	`  }`
	`}`
		3

Question	Answer	Marks

2(e)	e.g. Python
	`class PuzzleBox(HiddenBox):`
	`  # __PuzzleText String`
	`  # __Solution String`
	`def __init__(self, NewBoxName, NewCreator, NewDateHidden, NewLocation, NewPuzzleText, NewSolution):`
	`  super().__init__(NewBoxName, NewCreator, NewDateHidden, NewLocation)`
	`  self.__PuzzleText = NewPuzzleText`
	`  self.__Solution = NewSolution`	

Question	Answer	Marks

3(a)	1 mark per bullet to max 3
	• Declaring an array named QueueData with 20 values (As String)
	• Declaring a Start Pointer pointing to 0/1
	• Declaring an End Pointer pointing to 0/1
	
	e.g. VB.NET
	`Sub Main()`
	`  Dim QueueData(0 To 19) As String`
	`  Dim StartPointer As Integer = 0`
	`  Dim EndPointer As Integer = 0`
	`End Sub`
	
	e.g. Java
	`public static void main(String[] args){`
	`  String[] QueueData = new String[20];`
	`  int StartPointer = 0;`
	`  int EndPointer = 0;`
	`}`
	
	e.g. Python
	`QueueData = ["" for I in range(0, 20)]`
	`StartPointer = 0`
	`EndPointer = 0`
		3

Question	Answer	Marks

3(b)	1 mark per bullet to max 6
	• Declaring a function, taking data as a string parameter
	• Checking if the queue is full
	• … returning False
	• Adding the data to the array
	• … incrementing the end pointer
	• … returning True
	
	e.g. VB.NET
	`Function Enqueue(ByVal DataToAdd As String, ByRef QueueData() As String, ByRef EndP As Integer)`
	`    If EndP = 20 Then`
	`        Return False`
	`    Else`
	`        Queue(EndP) = DataToAdd`
	`        EndP += 1`
	`        Return True`
	`    End If`
	`End Function`
	
	e.g. Java
	`public static Boolean Enqueue(String DataToAdd, String[] QueueData, String EndP){`
	`    if(EndP == 20){`
	`        return false;`
	`    }else{`
	`        QueueData[EndP] = DataToAdd;`
	`        EndP = EndP + 1;`
	`        return true;`
	`    }`
	`}`
		6

Question	Answer	Marks

3(b)	e.g. Python
	`def Enqueue(DataToAdd, QueueData, EndP):`
	`  if(EndP == 20):`
	`  return False, EndP`
	`  else:`
	`  QueueData[EndP] = DataToAdd`
	`  EndP = EndP + 1`
	`  return True, EndP`	

Question	Answer	Marks
3(c)	1 mark per bullet to max 8 • Function header and inputs a filename • Checking if the file exists • … returning a –1 if not • Opening the file for reading • Looping until the QueueData is full OR there are no values left in the file • Reads line from the text file • … calls function from **part 3(b)** with the data as parameter … • … stores return value • Returns 1 if the queue was full • Returns 2 if all elements were added • Efficient use of parameter passing i.e. not using global variables e.g. VB.NET	**8**

```
Function ReadFile(ByRef QueueData() As String, ByRef StartP As Integer, ByRef EndP As
                                                                                Integer)
    Dim DataToInsert As String
    Dim filename As String
    Console.Writeline("Enter a filename")
    filename = Console.Readline()
    If System.IO.File.Exists(filename) Then
        Dim FileReader As New System.IO.StreamReader(filename)

        Dim Flag As Boolean = True
        While Flag = True And FileReader.Peek <> -1
            DataToInsert = FileReader.ReadLine()
            Flag = Enqueue(DataToInsert, QueueData, StartP, EndP)
        End While

        If Flag = False Then
            FileReader.Close()
            Return 1
        Else
            FileReader.Close()
            Return 2
        End If
```

Question	Answer	Marks

3(c)	Else
	Return -1
	End If
	End Function
	
	e.g. Java
	`public static int ReadFile(String[] QueueData, StartP, EndP){`
	`Scanner scanner = new Scanner(System.in);`
	`System.out.println("Enter a filename");`
	`String FileName = scanner.nextLine();`
	`File f = new File(FileName);`
	`if(f.exists()){`
	`try{`
	`BufferedReader reader = new BufferedReader(new FileReader(FileName));`
	`DataToInsert = reader.readLine();`
	`Boolean Flag = True`
	`while(Flag == True && DataToInsert != null)){`
	`DataToInsert = reader.readLine();`
	`Flag = Enqueue(DataToInsert, QueueData, EndP);`
	`}`
	`if (Flag == False){`
	`reader.close();`
	`return 1;`
	`}else{`
	`reader.close()`
	`return 2`
	`}`
	`}`
	`catch(IOException e){`
	`return -1;`
	`}`
	`}else{ return -1;`
	`}`	

Question	Answer	Marks

3(c)	e.g. Python
	`def ReadFile(QueueData, StartP, EndP):`
	`  FileName = input("Enter a filename")`
	`  if(os.path.isfile(FileName)):`
	`  f = open(FileName, "r")`
	`  Flag = True`
	`  DataToInsert = (f.readline()).strip()`
	`  while(Flag == True and DataToInsert != null):`
	`  Flag, EndP = Enqueue(QueueData, EndP)`
	`  DataToInsert = (f.readline()).strip()`
	`  if(Flag == False):`
	`  f.close()`
	`  return 1, EndP`
	`  else:`
	`  f.close()`
	`  return 2, EndP`
	`  else:`
	`  return -1, EndP`	

Question	Answer	Marks

3(d)(i)	1 mark per bullet
	• Calling the function from **part 3(c)** with appropriate parameters
	• Storing the return value from the function call
	• Outputting an appropriate message if the text file was not found
	• Outputting an appropriate message if the queue was full and outputting an appropriate message if all items were added to the queue
	
	e.g. VB.NET
	`Sub Main()`
	`  Dim QueueData(0 To 19) As String`
	`  Dim StartPointer As Integer = 0`
	`  Dim EndPointer As Integer = 0`
	
	`  Dim returnValue As Integer`
	
	`  returnValue = ReadFile(QueueData, StartPointer, EndPointer)`
	
	`  If returnValue = -1 Then`
	`  Console.WriteLine("The file could not be found")`
	`  ElseIf returnValue = 1 Then`
	`  Console.WriteLine("The queue was full, not all items were read")`
	`  Else`
	`  Console.WriteLine("All items successfully added")`
	`  End If`
	
	`End Sub`
		4

Question	Answer	Marks

3(d)(i)	e.g. Java
	`int ReturnValue;`
	`ReturnValue = readFile(QueueData, StartPointer, EndPointer);`
	`if (ReturnValue == -1){`
	`  System.out.println("The file could not be found")`
	`}elseif(ReturnValue == 1){`
	`  System.out.println("The queue was full, not all items were read")`
	`}else{`
	`  System.out.println("All items successfully added")`
	`}`
	
	e.g. Python
	`ReturnValue, EndPointer = ReadFile(QueueData, StartPointer, EndPointer):`
	`if(ReturnValue == -1):`
	`  print("The file could not be found")`
	`elif(ReturnValue == 1):`
	`  print("The queue was full, not all items were read")`
	`else:`
	`  print("All items successfully added")`	

3(d)(ii)	1 mark per bullet • `DataToAdd.txt` as outputting 'All items successfully added' • `SecondData.txt` as outputting 'The queue was full …' • `ThirdData.txt` as outputting 'The file could not be found'	**3**

Question	Answer	Marks

3(e)	1 mark per bullet to max 5
	• Function declaration
	• Checking if QueueData has sufficient values
	• … returning `No Items` if not
	• Reading 2 values from QueueData
	• Incrementing `StartP` twice // adding 2 to `StartP`
	• Concatenating the two values
	• … returning the result
	
	e.g. VB.NET
	`Function Remove(ByRef QueueData() As String, ByRef StartP As Integer, ByRef EndP As Integer)`
	`  Dim Value1, Value2 As String`
	`  If EndP - StartP < 2 Then`
	`  Return "No Items"`
	`  Else`
	`  Value1 = QueueData(StartP)`
	`  StartP += 1`
	`  Value2 = QueueData(StartP)`
	`  StartP += 1`
	`  Return (Value1 * " " & Value2)`
	`  End If`
	`End Function`
	
	e.g. Java
	`public static String remove(String[] QueueData, Startp, Endp){`
	`  String Value1, Value2`
	`  if(EndP - StartP < 2){`
	`  return "No items";`
	`  }else{`
	`  Value1 = QueueData[StartP];`
	`  StartP++;`
	`  Value2 = QueueData[StartP];`
	`  StartP++;`
	`  return (Value1 + " " + Value2);`
	`  }`
	`}`
		5

Question	Answer	Marks

3(e)	e.g. Python
	`def Remove(QueueData, StartP, EndP):`
	`  if(EndP - StartP < 2):`
	`  return "No Items", StartP, EndP`
	`  else:`
	`  Value1 = QueueData[StartP]`
	`  StartP = StartP + 1`
	`  Value2 = QueueData[StartP]`
	`  StartP = StartP + 1`
	`  return(Value1 + " " + Value2), StartP, EndP`	

BLANK PAGE