CAMBRIDGE
International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/43**

Paper 4 Practical **May/June 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 May/June 2024 series for most Cambridge IGCSE, Cambridge International A and AS Level and Cambridge Pre-U components, and some Cambridge O Level components.

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 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)	1 mark for:
	• Declaration of (global) array with identifier `DataStored` (Integer and 20 spaces) **and** `NumberItems` (Integer)
	e.g.
	Java
	`public static Integer[] DataStored = new Integer[20];`
	`public static Integer NumberItems= 0;`
	
	VB.NET
	`Dim DataStored(19) As Integer`
	`Dim NumberStored As Integer = 0`
	
	Python
	`global DataStored #integer`
	`global NumberItems #Integer 20 items`
		1

Question	Answer	Marks

1(b)	1 mark each
	• Procedure heading (and close where appropriate) with **no** parameter.
	• Prompt/output of suitable message to request the input of the quantity of numbers **and** reading in quantity of numbers **and** storing/using …
	• … each input in next space in `DataStored`
	e.g.
	Java
	`public static void Initialise(){`
	`  Scanner scanner = new Scanner(System.in);`
	`  Integer Quantity = 0;`
	`  do{`
	`  System.out.println("How many numbers will you enter up to 20?");`
	`  Quantity = Integer.parseInt(scanner.nextLine());`
	`  }while(Quantity <= 0 || Quantity > 20);`
	`  for(Integer X = 0; X < Quantity; X++){`
	`  System.out.println("Enter number");`
	`  DataStored[NumberItems] = Integer.parseInt(scanner.nextLine());`
	`  NumberItems++;`
	
	VB.NET
	`Sub Initialise()`
	`  Console.WriteLine("How many numbers will you enter?")`
	`  Dim Quantity As Integer`
	`  Do`
	`  Quantity = Console.ReadLine()`
	`  Loop Until (Quantity > 0 And Quantity < 21)`
	`  For Count = 0 To Quantity - 1`
	`  Console.WriteLine("Enter number")`
	`  DataStored(NumberStored) = Console.ReadLine()`
	`  NumberStored += 1`
	`  Next`
	`End Sub`
		5

Question	Answer	Marks

1(b)	Python
	`def Initialise():`
	`  global DataStored`
	`  global NumberItems`
	`  Valid = False`
	`  while(Valid == False):`
	`  NumberItems = int(input("How many numbers will you enter?")) #loop until < 20`
	`  if NumberItems > 0 and NumberItems< 21:`
	`  Valid = True`
	`  for Count in range(0, NumberItems):`
	`  DataStored.append(int(input("Enter number")))`	

1(c)(i)	1 mark each:
	• Storing `0` in `NumberItems` **and** then calling `Initialise()`
	• Outputting all contents of array `DataStored`
	e.g.
	Java
	`public static Integer NumberItems= 0;`
	`Initialise();`
	`for(Integer X = 0; X < NumberItems; X++){`
	`  System.out.println(DataStored[X]);`
	
	VB.NET
	`NumberItems = 0`
	`Initialise()`
	`For X = 0 To NumberItems - 1`
	`  Console.WriteLine(DataStored(X))`
	`Next`
	
	Python
	`NumberItems = 0`
	`Initialise()`
	`print(DataStored)`
		2

Question	Answer	Marks
1(c)(ii)	1 mark each • Output showing quantity entered twice (30 and 5) with first being invalid • Array output 3 9 4 1 2 e.g. How many numbers will you enter?30 How many numbers will you enter?5 Enter number3 Enter number9 Enter number4 Enter number1 Enter number2 [3, 9, 4, 1, 2]	**2**

Question	Answer	Marks

1(d)(i)	1 mark each
	• Procedure header (and end where appropriate) **and** looping through each array element
	• Working **inner** loop …
	• …comparison of elements…
	• …swapping of elements
	e.g.
	Java
	`public static void BubbleSort(){`
	`  Integer Temp = 0;`
	`  for(Integer Count = 0; Count < NumberItems; Count++){`
	`  for(Integer Count2 = 0; Count2 < NumberItems - 1; Count2++){`
	`  if(DataStored[Count2] > DataStored[Count]){`
	`  Temp = DataStored[Count2];`
	`  DataStored[Count2] = DataStored[Count];`
	`  DataStored[Count] = Temp;`
	`  }`
	`  }`
	`  }`
	`}`
	VB.NET
	`Sub BubbleSort()`
	`  Dim Temp As Integer`
	`  For Count = 0 To NumberStored - 1`
	`  For Count2 = 0 To NumberStored - 2`
	`  If (DataStored(Count2) > DataStored(Count)) Then`
	`  Temp = DataStored(Count)  DataStored(Count) = DataStored(Count2)`
	`  DataStored(Count2) = Temp`
	`  End If`
	`  Next`
	`  Next`
	`End Sub`
		4

Question	Answer	Marks

1(d)(i)	Python
	`def BubbleSort():`
	`  global DataStored`
	`  global NumberItems`
	`  for Count in range(0, NumberItems):`
	`  for Count2 in range(0, NumberItems-1):`
	`  if DataStored[Count2] > DataStored[Count]:`
	`  DataStored[Count2], DataStored[Count] = DataStored[Count], DataStored[Count2]`	

1(d)(ii)	1 mark for calling `BubbleSort()` **and** outputting array contents after
	e.g.
	VB.NET
	`BubbleSort()`
	`For X = 0 To NumberStored - 1`
	`  Console.WriteLine(DataStored(X))`
	`Next`
	
	e.g. Java
	`BubbleSort();`
	`for(Integer X = 0; X < NumberItems; X++){`
	`  System.out.println(DataStored[X]);`
	`}`
	e.g. Python
	`BubbleSort()`
	`print(DataStored)`
		1

1(d)(iii)	1 mark for screenshot showing the inputs and the values in the correct order
	e.g.
	`How many numbers will you enter?5`
	`Enter number3`
	`Enter number9`
	`Enter number4`
	`Enter number1`
	`Enter number2`
	`[1, 2, 3, 4, 9]`
		1

Question	Answer	Marks

1(e)(i)	1 mark each
	• Function header `BinarySearch` taking `DataToFind` as a parameter
	• Calculating the mid value `(First + Last) \ 2` or equivalent **inside loop**
	• Checking if the data at `mid` is the parameter **and** returning `mid` **inside loop**
	• If `DataToFind < mid`, updating `Last/Upper` with `mid – 1` **inside loop**
	• If `DataToFind > mid`, updating `First/Lower` with `mid + 1` **inside loop**
	• Returning `–1` when not found **and** a suitable loop with end criteria
	e.g.
	Java
	`public static Integer BinarySearch(Integer DataToFind){`
	`  Integer MidValue = 0;`
	`  Integer First = 0;`
	`  Integer Last = NumberItems;`
	`  while (First <= Last){`
	`  MidValue = (First + Last) / 2;`
	`  if(DataToFind == DataStored[MidValue]){`
	`  return MidValue;`
	`  }`
	`  if(DataToFind < DataStored[MidValue]){`
	`  Last = MidValue - 1;`
	`  }else{`
	`  First = MidValue + 1;`
	`  }`
	`  }`
	`  return -1;`
	`}`
		6

Question	Answer	Marks

1(e)(i)	VB.NET
	`Function BinarySearch(DataToFind)`
	`    Dim First As Integer = 0`
	`    Dim Last As Integer = NumberItems`
	`    Dim MidValue As Integer`
	`    While (First <= Last)`
	`        MidValue = (First + Last) / 2`
	`        If DataToFind = DataStored(MidValue) Then`
	`            Return MidValue`
	`        End If`
	`        If DataToFind < DataStored(MidValue) Then`
	`            Last = MidValue - 1`
	`        Else`
	`            First = MidValue + 1`
	`        End If`
	`    End While`
	`    Return -1`
	`End Function`
	
	Python
	`def BinarySearch(DataToFind):`
	`    global DataStored`
	`    global NumberItems`
	`    First = 0`
	`    Last= NumberItems`
	`    while(First <= Last):`
	`        MidValue = int((First + Last) / 2)`
	
	`        if DataToFind == DataStored[MidValue]:`
	`            return MidValue`
	`        if DataToFind < DataStored[MidValue]:`
	`            Last = MidValue - 1`
	`        else:`
	`            First = MidValue + 1`
	
	`    return -1`	

Question	Answer	Marks

1(e)(ii)	1 mark each:
	• Taking number as input
	… calling `BinarySearch` with input
	• Outputting value returned
	e.g.
	Java
	`Scanner scanner = new Scanner(System.in);`
	`System.out.println("Enter a number to find");`
	`Integer Search = Integer.parseInt(scanner.nextLine());`
	`System.out.println(BinarySearch(Search));`
	
	VB.NET
	`Console.WriteLine("Enter a number to find")`
	`Dim Search As Integer = Console.ReadLine()`
	`Console.WriteLine(BinarySearch(Search))`
	
	Python
	`Search = int(input("Enter a number to find"))`
	`print(BinarySearch(Search))`
		3

Question	Answer	Marks

1(e)(iii)	1 mark for each test
	e.g.
	Test 1 – Accept found in index 16
	`How many numbers will you enter?5`
	`Enter number1`
	`Enter number6`
	`Enter number2`
	`Enter number8`
	`Enter number10`
	`[1, 2, 6, 8, 10]`
	`Enter a number to find2`
	`1`
	
	Test 2
	`How many numbers will you enter?5`
	`Enter number1`
	`Enter number6`
	`Enter number2`
	`Enter number8`
	`Enter number10`
	`[1, 2, 6, 8, 10]`
	`Enter a number to find7`
	`-1`
		2

Question	Answer	Marks

2(a)(i)	1 mark each to max 4
	• Class `Tree` declaration (and end where appropriate)
	• All 5 attributes declared as private with correct identifiers and data types
	• Constructor header (and end) taking 5 parameters
	• Constructor assigns parameters to attributes
	e.g.
	Java
	`class Tree{`
	`  private String TreeName;`
	`  private Integer HeightGrowth;`
	`  private Integer MaxWidth;`
	`  private Integer MaxHeight;`
	`  private String Evergreen;`
	
	`  public Tree(String Name, Integer HGrowth, Integer MaxH, Integer MaxW, String PEvergreen){`
	`  TreeName = Name;`
	`  HeightGrowth = HGrowth;`
	`  MaxWidth = MaxW;`
	`  MaxHeight = MaxH;`
	`  Evergreen = PEvergreen;`
	`}}`
		4

Question	Answer	Marks

2(a)(i)	VB.NET
	`Class Tree`
	`  Private TreeName As String`
	`  Private HeightGrowth As Integer`
	`  Private MaxHeight As Integer`
	`  Private MaxWidth As Integer`
	`  Private Evergreen As String`
	`  Sub New(Name, HGrowth, MaxH, MaxW, PEvergreen)`
	`  TreeName = Name`
	`  HeightGrowth = HGrowth`
	`  MaxHeight = MaxH`
	`  MaxWidth = MaxW`
	`  Evergreen = PEvergreen`
	`  End Sub`
	`End Class`
	
	Python
	`class Tree:`
	`  def __init__(self, Name, HGrowth, MaxH, MaxW, PEvergreen):`
	`  self.__TreeName = Name`
	`  self.__HeightGrowth = HGrowth`
	`  self.__MaxHeight = MaxH`
	`  self.__MaxWidth = MaxW`
	`  self.__Evergreen = PEvergreen`	

Question	Answer	Marks

2(a)(ii)	1 mark each
	• 1 get method with no parameter …
	• … returning correct attribute
	• Remaining 4 correct
	e.g.
	Java
	`public String GetTreeName(){`
	`  return TreeName;`
	`}`
	`public Integer GetGrowth(){`
	`  return HeightGrowth;`
	`}`
	`public Integer GetMaxWidth(){`
	`  return MaxWidth;`
	`}`
	`public Integer GetMaxHeight(){`
	`  return MaxHeight;`
	`}`
	`public String GetEvergreen(){`
	`  return Evergreen;`
	`}`
		3

Question	Answer	Marks

2(a)(ii)	VB.NET
	`Function GetTreeName()`
	`    Return TreeName`
	`End Function`
	`Function GetMaxHeight()`
	`    Return MaxHeight`
	`End Function`
	`Function GetMaxWIdth()`
	`    Return MaxWidth`
	`End Function`
	`Function GetGrowth()`
	`    Return HeightGrowth`
	`End Function`
	`Function GetEvergreen()`
	`    Return Evergreen`
	`End Function`
	
	Python
	`def GetTreeName(self):`
	`    return self.__TreeName`
	`def GetMaxHeight(self):`
	`    return self.__MaxHeight`
	`def GetMaxWidth(self):`
	`    return self.__MaxWidth`
	`def GetGrowth(self):`
	`    return self.__HeightGrowth`
	`def GetEvergreen(self):`
	`    return self.__Evergreen`	

2(b)	1 mark for: • appropriate use of exception handling, with catch and output 1 mark each to max 6 • Function header (and end where appropriate) **and** declaration of array (of type `Tree` with min 9 elements) • Opening text file `Trees.txt` to read **and** closing the file • Reading each line of text (until EOF, or 9 times) • Splitting each line into the 5 elements … casting height growth, max height and max width to integers … creating a new object of type `Tree` with the 5 values … storing each object in the array **and** returning the array e.g. Java	**7**

```
public static Tree[] ReadData(){
      String TextFile = "Trees.txt";
      String[] TempData = new String[5];
      Tree[] TreeData = new Tree[20];
      String Line;
      try{
            FileReader f = new FileReader(TextFile);
            BufferedReader Reader = new BufferedReader(f);
            for(Integer X = 0; X < 9; X++){
                  try{
                        Line = Reader.readLine();
                        TempData = Line.split(",");
                        TreeData[X] = new Tree(TempData[0], Integer.parseInt(TempData[1]),
Integer.parseInt(TempData[2]), Integer.parseInt(TempData[3]), TempData[4]);
                  }catch(IOException ex){}
            }
            try{
                  Reader.close();
            }catch(IOException ex){}

      }catch(FileNotFoundException e){
            System.out.println("File not found");
      }
      return TreeData;
}
```

Question	Answer	Marks

2(b)	VB.NET
	`Function ReadData()`
	`Dim TreeObjects(10) As Tree`
	`Dim TextFile As String = "Trees.txt"`
	`try`
	`Dim FileReader As New System.IO.StreamReader(TextFile)`
	`Dim TreeData(10) As String`
	`Dim TreeSplit() As String`
	`For Count = 0 To 8`
	`TreeData(Count) = FileReader.ReadLine()`
	`Next Count`
	`FileReader.Close()`
	`For X = 0 To 8`
	`TreeSplit = TreeData(X).Split(",")`
	`TreeObjects(X) = New Tree(TreeSplit(0), Integer.Parse(TreeSplit(1)), Integer.Parse(TreeSplit(2)), Integer.Parse(TreeSplit(3)), TreeSplit(4))`
	`Next X`
	`Catch ex As Exception`
	`Console.WriteLine ("invalid file")`
	`End Try`
	`Return TreeObjects`
	`End Function`	

Question	Answer	Marks

2(b)	Python
	`def ReadData():`
	`    TreeObjects=[]`
	`    try:`
	`        File = open("Trees.txt")`
	`        TreeData = []`
	`        TreeData = File.read().split("\n")`
	`        SplitTrees = []`
	`        for Item in TreeData:`
	`            SplitTrees.append(Item.split(","))`
	`        File.close()`
	`        for Item in SplitTrees:`
	
	`TreeObjects.append(Tree(Item[0],int(Item[1]),int(Item[2]),int(Item[3]),Item[4]))`
	`    except IOError:`
	`        print ("invalid file")`
	`    return TreeObjects`	

Question	Answer	Marks

2(c)	1 mark each
	• Procedure heading (and end) taking **one** parameter (of type `Tree`)
	and using get methods to access tree name, height, width, growth
	• Outputs all 4 attributes (`TreeName`, `MaxHeight`, `MaxWidth`, `GetGrowth`)
	• Checks if it is evergreen…
	… correct messages are output if evergreen and otherwise
	
	e.g.
	Java
	`public static void PrintTrees(Tree TreeItem){`
	`String Final = "does not lose its leaves";`
	`if((TreeItem.GetEvergreen()).compareTo("No") == 0){`
	`Final = "loses its leaves each year";`
	`}`
	`System.out.println(TreeItem.GetTreeName() + " has a maximum height " + TreeItem.GetMaxHeight() + " a maximum width " + TreeItem.GetMaxWidth() + " and grows " + TreeItem.GetGrowth() + " cm a year. It " + Final);`
	`}`
	
	VB.NET
	`Sub PrintTrees(Item)`
	`Dim Final As String = "does not lose its leaves"`
	`If (Item.GetEvergreen() = "No") Then`
	`Final = "loses its leaves each year"`
	`End If`
	`Console.WriteLine(Item.GetTreeName() & " has a maximum height " & Item.GetMaxHeight() & " a maximum width " & Item.GetMaxWidth() & " and grows " & Item.GetGrowth() & "cm a year. It" & Final)`
	`End Sub`
		4

Question	Answer	Marks

2(c)	Python
	`def PrintTrees(Item):`
	
	`    Final = "does not lose its leaves"`
	`    if Item.GetEvergreen() == "No":`
	`        Final = "loses its leaves each year"`
	`    print(Item.GetTreeName(), "has a maximum height", Item.GetMaxHeight(),"a maximum width",Item.GetMaxWidth(),"and grows", Item.GetGrowth(),"cm a year. It",Final)`	

2(d)(i)	• 1 mark each
	• Calling `ReadData()` **and** storing/using return value (as array of type `Tree`)…
	…calling `PrintTrees()` with first object in returned array as parameter
	
	e.g.
	Java
	`Tree[] TreeData = new Tree[20];`
	`TreeData = ReadData();`
	`PrintTrees(TreeData[0]);`
	
	VB.NET
	`Sub Main(args As String())`
	`        Dim TreeObjects(10) As Tree`
	`        TreeObjects = ReadData()`
	`        PrintTrees(Treeobjects(0))`
	`End Sub`
	
	Python
	`TreeObjects = ReadData()`
	`PrintTrees(TreeObjects[0])`
		2

2(d)(ii)	Screenshot showing output `Beech has a maximum height 400 a maximum width 200 and grows 30 cm a year. It loses its leaves each year`	**1**

Question	Answer	Marks

2(e)(i)	1 mark each to max 6
	• Procedure header (and close) taking array of `Tree` objects as a parameter **and** reading evergreen, max height and max width **once** as input from the user
	
	• Looping through each array object …
	• … comparing each width input >= `MaxWidth`, height input >= `MaxHeight`
	• … comparing each evergreen input with `Evergreen` … **when all true** (all requirements met) - appending object in new array
	• Calling `PrintTrees()` with each valid object
	• Outputting suitable message if no trees appropriate
	
	e.g.
	Java
	`public static void ChooseTree(Tree[] Trees){`
	`  Scanner scanner = new Scanner(System.in);`
	`  System.out.println("Do you want a tree that loses its leaves (enter lose), or keeps its leaves (enter keep)") ;`
	`  String Evergreen = (scanner.nextLine());`
	`  System.out.println("What is the maximum tree height in cm");`
	`  Integer MaxHeight = Integer.parseInt(scanner.nextLine());`
	`  System.out.println("What is the maximum tree width in cm");`
	`  Integer MaxWidth = Integer.parseInt(scanner.nextLine());`
	`  Tree[] Options = new Tree[20];`
	`  String keep;`
	`  Tree Selected;`
	`  Boolean Valid = false;`
	`  if(((Evergreen.toLowerCase()).compareTo("keep") == 0) || ((Evergreen.toLowerCase()).compareTo("keep leaves") == 0) || ((Evergreen.toLowerCase()).compareTo("keeps its leaves") == 0)){`
	`  keep = "Yes";`
	`  }else{`
	`  keep = "No";`
	`  }`
	`  Integer Counter = 0;`
	`  for(Integer X = 0; X < 9; X++){`
		6

Question	Answer	Marks

2(e)(i)	`if((Trees[X].GetMaxHeight() <= MaxHeight) && (Trees[X].GetMaxWidth() <= MaxWidth) && (keep.compareTo(Trees[X].GetEvergreen())==0)){`
	`Options[Counter] = Trees[X];`
	`PrintTrees(Trees[X]);`
	`Counter = Counter + 1;`
	`}`
	`}`
	`if(Counter == 0){`
	`System.out.println("No suitable trees");`
	`}`
	`}`
	
	VB.NET
	`Sub ChooseTree(Trees)`
	`Console.WriteLine("Do you want a tree that loses its leaves (enter lose), or keeps its leaves (enter keep)")`
	`Dim Evergreen As String = Console.ReadLine()`
	`Console.WriteLine("What is the maximum tree height in cm")`
	`Dim MaxHeight As Integer = Console.ReadLine()`
	`Console.WriteLine("What is the maximum tree width in cm")`
	`Dim MaxWidth As Integer = Console.ReadLine()`
	`Dim Options(0 To 9) As Tree`
	`Dim keep As String`
	`Dim Valid As Boolean`
	`Dim Selected As Tree`
	`If Evergreen.ToLower() = "keep" Or Evergreen.ToLower() = "keep leaves" Or Evergreen.ToLower() = "keeps its leaves" Then`
	`keep = "Yes"`
	`Else`
	`keep = "No"`	

Question	Answer	Marks
2(e)(i)		

```
        End If
        Dim count As Integer = 0
        For x = 0 To 8
            If Trees(x).GetMaxHeight() <= MaxHeight And Trees(x).GetMaxWidth() <= MaxWidth
And keep = Trees(x).GetEvergreen() Then
                Options(count) = Trees(x)
                PrintTrees(Trees(x))
                count = count + 1
            End If
        Next x
        If count = 0 Then
            Console.WriteLine("No suitable trees")
        End If
    End Sub
```

Python

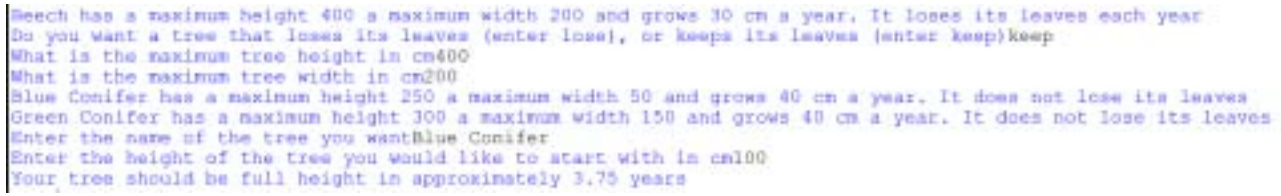
```
def ChooseTree(Trees):
    Evergreen = input("Do you want a tree that loses its leaves (enter lose), or keeps its
leaves (enter keep)")
    MaxHeight = int(input("What is the maximum tree height in cm"))
    MaxWidth = int(input("What is the maximum tree width in cm"))
    Options = []
    if Evergreen.lower() == "keep" or Evergreen.lower() == "keep leaves" or
Evergreen.lower() == "keeps its leaves":
        keep = "Yes"
    else:
        keep = "No"
    for Item in Trees:

        if Item.GetMaxHeight() <= MaxHeight and Item.GetMaxWidth() <= MaxWidth and keep ==
Item.GetEvergreen():
            Options.append(Item)
            PrintTrees(Item)
    if len(Options) == 0:
        print("No suitable trees")
```

Question	Answer	Marks

2(e)(ii)	1 mark each to max
	• Taking tree name and initial height as input
	• Finding the tree, calculating **and** outputting the number of years to get to maximum height
	VB.NET
	`Valid = False`
	`Dim Start As Integer`
	`Dim Years As Single`
	`Dim Choice As String`
	`While Valid = False`
	`  Console.WriteLine("Enter the name of the tree you want")`
	`  Choice = Console.ReadLine()`
	`  For X = 0 To count - 1`
	`  If Options(X).GetTreeName() = Choice Then`
	`  Valid = True`
	`  Selected = Options(X)`
	`  Console.WriteLine("Enter the height of the tree you would like to start with in cm")`
	`  Start = Console.ReadLine()`
	`  Years = (Selected.GetMaxHeight() - Start) / Selected.GetGrowth()`
	`  Console.WriteLine("Your tree should be full height in approximately " & Years & " years")`
	`  End If`
	`  Next X`
	`End While`
		2

Question	Answer	Marks

2(e)(ii)	Java
	`Integer Start;`
	`Float Height;`
	`Float Growth;`
	`Float Years;`
	`while(Valid == false){`
	`  System.out.println("Enter the name of the tree you want");`
	`  String Choice = scanner.nextLine();`
	`  for(Integer X = 0; X < Counter; X++){`
	`  if((Options[X].GetTreeName()).compareTo(Choice)==0){`
	`  Valid = true;`
	`  Selected = Options[X];`
	`  System.out.println("Enter the height of the tree you would like to start with in cm");`
	`  Start = Integer.parseInt(scanner.nextLine());`
	`  Height = (Selected.GetMaxHeight()).floatValue();`
	`  Growth = (Selected.GetGrowth()).floatValue();`
	`  Years = (Height - Start) / Growth;`
	`  System.out.println("Your tree should be full height in approximately "+ Years + " years");`
	`  }`
	`  }`
	`}`
	
	Python:
	`Valid = False`
	`while Valid == False:`
	`  Choice = input("Enter the name of the tree you want")`
	`  for Item in Options:`
	`  if Item.GetTreeName() == Choice:`
	`  Valid = True`
	`  Selected = Item`
	`  Start = int(input("Enter the height of the tree you would like to start with in cm"))`
	`  Years = (Selected.GetMaxHeight() - Start)/Selected.GetGrowth()`
	`  print("Your tree should be full height in approximately", Years,"years")`	

Question	Answer	Marks
2(e)(iii)	1 mark each • Screenshot shows the user requirements input (height 400, width 200, evergreen) **and** outputs the correct trees (Blue conifer and green conifer) • Screenshot shows the tree selection input (Blue Conifer with height 100) **and** outputs the correct result (3 years / 3.75 / 4 years)  Beech has a maximum height 400 a maximum width 200 and grows 30 cm a year. It loses its leaves each year Do you want a tree that loses its leaves (enter lose), or keeps its leaves (enter keep)keep What is the maximum tree height in cm400 What is the maximum tree width in cm200 Blue Conifer has a maximum height 250 a maximum width 50 and grows 40 cm a year. It does not lose its leaves Green Conifer has a maximum height 300 a maximum width 150 and grows 40 cm a year. It does not lose its leaves Enter the name of the tree you wantBlue Conifer Enter the height of the tree you would like to start with in cm100 Your tree should be full height in approximately 3.75 years	**2**

Question	Answer	Marks

3(a)	1 mark each
	• `QueueData` as 1D (string) array initialised to 20 null values
	and `QueueHead` initialised to `-1`, `QueueTail` initialised to `-1`
	e.g.
	Java
	`class Queue{`
	`  public static String[] QueueData = new String[20];`
	`  public static Integer QueueHead;`
	`  public static Integer QueueTail;`
	`  public static void main(String args[]){`
	`  for(Integer x = 0; x < 20; x++){`
	`  QueueData[x] = "";`
	`  }`
	`  QueueHead = -1;`
	`  QueueTail = -1;`
	`  }`
	`}`
	VB.NET
	`Dim QueueData(0 To 20) As String`
	`  Dim QueueHead As Integer = -1`
	`  Dim QueueTail As Integer = -1`
	`Sub Main(args As String())`
	`  For x = 0 To 19`
	`  QueueData(x) = ""`
	`  Next`
	`End Sub`
	
	Python
	`global QueueData`
	`global QueueHead`
	`global QueueTail`
	`QueueData = []`
	`for x in range(0, 20):`
	`  QueueData.append("")`
	`QueueHead = -1`
	`QueueTail = -1`
		1

Question	Answer	Marks

3(b)	1 mark each
	• Function header (and end) taking **one** parameter **and** returns a Boolean value in all instances
	• Checks if queue is full **and** returns `FALSE`
	• (If not full) Inserts data item to `QueueTail + 1`
	and increments `QueueTail`
	and returns `TRUE`
	• Assigns `QueueHead` to `0` when first element is entered (this can come from incrementing)
	e.g.
	Java
	`public static Boolean Enqueue(String DataToInsert){`
	`  if(QueueTail == 19){`
	`  return false;`
	`  }else if(QueueHead == -1){`
	`  QueueHead = 0;`
	`  }`
	`  QueueTail = QueueTail + 1;`
	`  QueueData[QueueTail] = DataToInsert.substring(0,6);`
	`  return true;`
	`}`
	VB.NET
	`Function Enqueue(ByVal DataToInsert)`
	`  If QueueTail = 19 Then`
	`  Return False`
	`  ElseIf QueueHead = -1 Then`
	`  QueueHead = 0`
	`  End If`
	`  QueueTail = QueueTail + 1`
	`  QueueData(QueueTail) = DataToInsert`
	`  Return True`
	`End Function`
		4

Question	Answer	Marks

3(b)	Python
	`def Enqueue(DataToInsert):`
	`    global QueueData`
	`    global QueueHead`
	`    global QueueTail`
	`    if QueueTail == 19:`
	`        return False`
	`    elif QueueHead == -1:`
	`        QueueHead = 0`
	`    QueueTail = QueueTail + 1`
	`    QueueData.append(DataToInsert)`
	`    return True`	

Question	Answer	Marks

3(c)	1 mark each
	• `Dequeue` function header (and end) returning a **string** in all cases
	• Check if queue is empty
	and return `"false"`
	• (otherwise) remove value at `QueueHead`
	and increment `QueueHead`
	and return value from array
	e.g.
	Java
	`public static String Dequeue(){`
	`  if(QueueHead < 0 || QueueHead > 20 || QueueHead > QueueTail){`
	`  return "false";`
	`  }`
	`  QueueHead++;`
	`  return QueueData[QueueHead-1];`
	`}`
	VB.NET
	`Function Dequeue()`
	`  If QueueHead < 0 Or QueueHead > 20 Or QueueHead > QueueTail Then`
	`  Return "false"`
	`  Else`
	`  QueueHead = QueueHead + 1`
	`  Return QueueData(QueueHead - 1)`
	`  End If`
	`End Function`
		3

Question	Answer	Marks

3(c)	Python
	`def Dequeue():`
	`  global QueueData`
	`  global QueueHead`
	`  global QueueTail`
	`  if QueueHead < 0 or QueueHead > 20 or QueueHead > QueueTail:`
	`  return False`
	`  else:`
	`  QueueHead = QueueHead + 1`
	
	`  return QueueData[QueueHead-1]`	

3(d)(i)	1 mark each to max 6
	• `StoreItems` header (function/procedure and end where appropriate) **and** takes 10 inputsi
	• Input is split and first 6 characters used in calculation (as integers) …
	• … multiplication by 1 and 3 alternately, adding to total, dividing by 10, rounding down/cast int …
	• … comparing check digit to character in position 6
	• … including comparison of X for 10
	
	• Calling `Enqueue` with first 6 characters **when valid**
	• … outputting appropriate message on return (for both inserted and queue full)
	• Counts **and** outputs number of invalid inputs
	e.g.
	Java
	`public static void StoreItems(){`
	`  Integer Count = 0;`
	`  Integer Total = 0;`
	`  String Data;`
	`  Boolean Result;`
	`  Scanner scanner = new Scanner(System.in);`
	`  for(Integer X = 0; X < 10; X++){`
	`  System.out.println("Enter data");`
	`  Data = scanner.nextLine();`
	`  Total = Integer.parseInt(Data.substring(0,1)) +`
		6

Question	Answer	Marks
3(d)(i)		

```
Integer.parseInt(Data.substring(1,2)) * 3 + Integer.parseInt(Data.substring(2,3)) +
Integer.parseInt(Data.substring(3,4)) * 3 + Integer.parseInt(Data.substring(4,5)) +
Integer.parseInt(Data.substring(5,6)) * 3;
           Total = Total / 10;
           if((Total == 10 && Data.substring(6).compareTo("X")==0)){
                Result = Enqueue(Data);
                if(Result == true){
                     System.out.println("Inserted item");
                }else{
                     System.out.println("Queue full");
                }
           }else if(Total == Integer.parseInt(Data.substring(6,7))){
                Result = Enqueue(Data);
                if(Result == true){
                     System.out.println("Inserted item");
                }else{
                     System.out.println("Queue full");
                }
           }else{
                Count = Count + 1;
           }
      }

           System.out.println("There were " + Count + " invalid items");
}
```

VB.NET

```
 Sub StoreItems()

    Dim Count As Integer = 0
    Dim Total As Integer = 0
    Dim Data As String
    Dim Result As Boolean
    For X = 0 To 9
        Console.WriteLine("Enter data")
        Data = Console.ReadLine()
```

Question	Answer	Marks

3(d)(i)	`Total = Integer.Parse(Data.Substring(0, 1)) + Integer.Parse(Data.Substring(1, 1)) * 3 + Integer.Parse(Data.Substring(2, 1)) + Integer.Parse(Data.Substring(3, 1)) * 3 + Integer.Parse(Data.Substring(4, 1)) + Integer.Parse(Data.Substring(5, 1)) * 3`
	`Total = Total \ 10`
	`If (Total = 10 And Data.Substring(6, 1) = "X") Then`
	`Result = Enqueue(Data.Substring(0, 6))`
	`If Result = True Then`
	`Console.WriteLine("Inserted item")`
	`Else`
	`Console.WriteLine("Queue full")`
	`End If`
	`ElseIf Total = Integer.Parse(Data.Substring(6, 1)) Then`
	`Result = Enqueue(Data)`
	`If Result = True Then`
	`Console.WriteLine("Inserted item")`
	`Else`
	`Console.WriteLine("Queue full")`
	`End If`
	`Else`
	`Count = Count + 1`
	`End If`
	`Next`
	`Console.WriteLine("There were " & Count & " invalid items")`
	
	`End Sub`	

Question	Answer	Marks

3(d)(i)	Python
	`def StoreItems():`
	`  global QueueData`
	`  global QueueHead`
	`  global QueueTail`
	`  Count = 0`
	`  for X in range(0, 10):`
	`  Data = input("Enter data")`
	`  Total= int(Data[0]) + int(Data[1]) * 3 + int(Data[2]) + int(Data[3]) * 3 + int(Data[4]) + int(Data[5]) * 3`
	`  Total = int(Total / 10)`
	`  if((Total == 10 and Data[6] == "X") or (Total == int(Data[6]))):`
	`  Result = Enqueue(Data[0:6])`
	
	`  if(Result == True):`
	`  print("Inserted item")`
	`  else:`
	`  print("Queue full")`
	`  else:`
	`  Count = Count + 1`
	
	`  print("There were", Count,"Invalid items")`	

Question	Answer	Marks

3(d)(ii)	• Calling `StoreItems()`
	and `Dequeue()` **once**
	and outputting a suitable message if the queue was empty
	and outputting the returned value if the queue was not empty
	e.g.
	Java
	`public static void main(String args[]){`
	`  for(Integer x = 0; x < 20; x++){`
	`  QueueData[x] = "";`
	`  }`
	`  QueueHead = -1;`
	`  QueueTail = -1;`
	`  StoreItems();`
	`  String Value = Dequeue();`
	`  if(Value.compareTo("false") == 0){`
	`  System.out.println("No data items");`
	`  }else{`
	`  System.out.println("Item code " + Value);`
	`  }`
	`}`
	VB.NET
	`Sub Main(args As String())`
	`  For x = 0 To 19`
	`  QueueData(x) = ""`
	`  Next`
	`  StoreItems()`
	`  Dim ReturnValue As String = Dequeue()`
	`  If (ReturnValue = "false") Then`
	`  Console.WriteLine("No data items")`
	`  Else`
	`  Console.WriteLine("Item code " & ReturnValue)`
	`  End If`
	
	`  End Sub`
		1

Question	Answer	Marks

3(d)(ii)	Python
	`QueueData = []`
	`for x in range(0, 20):`
	`  QueueData.append("")`
	`QueueHead = -1`
	`QueueTail = -1`
	`StoreItems()`
	
	`Value = Dequeue()`
	`if Value == False:`
	`  print("No data items")`
	`else:`
	`  print("Item code", Value)`	

3(d)(iii)	1 mark each
	• Data input of 10 values **and** output a message saying there are 4 invalid items
	• 999999 output
	e.g.
	`Enter data999999X`
	`Inserted item`
	`Enter data1251484`
	`Inserted item`
	`Enter data5500212`
	`Inserted item`
	`Enter data0033585`
	`Enter data9845788`
	`Inserted item`
	`Enter data6666666`
	`Enter data3258746`
	`Enter data8111022`
	`Inserted item`
	`Enter data7568557`
	`Inserted item`
	`Enter data0012353`
	`There were 4 Invalid items`
	`Item code 999999`
		2