CAMBRIDGE
International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/42**

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 each to max 6
	
	• Procedure declaration (and end where appropriate) taking (string) parameter
	• Opening file using parameter filename to read…
	• ...using exception handling with try and catch and output
	• Reading in the data for each line in that file …
	• … and storing in (global) array
	• Removing carriage return from each line read in (Java, Python)
	• Counting the number of answers (excluding main word)
	• Closing the file (might be within the Python opening file statement)
	
	e.g.
	Java
	`public static String[] WordArray;`
	`public static Integer NumberWords;`
	`public static void ReadWords(String FileName){`
	`  try{`
	`  FileReader f = new FileReader(FileName);`
	`  try{`
	`  BufferedReader Reader = new BufferedReader(f);`
	`  String Line= Reader.readLine();`
	`  while (Line != null){`
	`  WordArray[NumberWords] = Line.replace("\n","");`
	`  NumberWords++;`
	`  Line = Reader.readLine();`
	`  }`
	`  Reader.close();`
	`  }catch(IOException ex){}`
	`  }catch(FileNotFoundException e){`
	`  System.out.println("File not found");`
	`  }`
	`}`
		6

Question	Answer	Marks

1(a)	VB.NET
	`Sub ReadWords(FileName As String)`
	`  Try`
	`  Dim DataReader As StreamReader = New StreamReader(FileName)`
	`  NumberWords = -1`
	`  Do Until DataReader.EndOfStream`
	`  WordArray(NumberWords) = DataReader.ReadLine()`
	`  NumberWords = NumberWords + 1`
	`  Loop`
	`  DataReader.Close()`
	`  Catch ex As Exception`
	`  Console.WriteLine("Invalid file")`
	`  End Try`
	`End Sub`
	
	Python
	`def ReadWords(FileName):`
	`  global WordArray`
	`  global NumberWords`
	`  try:`
	`  File = open(FileName, 'r')`
	`  DataRead = File.read().strip()`
	`  File.close()`
	`  WordArray = DataRead.split()`
	`  NumberWords = len(WordArray)`
	`  except:`
	`  print("Cannot read file")`	

Question	Answer	Marks

1(b)	1 mark each
	
	• Outputting appropriate message to ask user to enter choice
	• Taking input from user, storing/using the input value
	• Conversion of input to filename…
	• …calling `ReadWords()` with correct filename in each case
	
	e.g.
	Java
	`public static void main(String args[]){`
	`  WordArray = new String[100];`
	`  NumberWords = 0;`
	`  Scanner scanner = new Scanner(System.in);`
	`  System.out.println("Easy, medium or hard?");`
	`  String Choice = scanner.nextLine();`
	`  if(Choice.equals("Easy")){`
	`  ReadWords("Easy.txt");`
	`  }else if(Choice.equals("medium")){`
	`  ReadWords("Medium.txt");`
	`  }else{`
	`  ReadWords("Hard.txt");`
	`  }`
	`}`
	
	VB.NET
	`Sub Main(args As String())`
	`  Console.WriteLine("Easy, medium or hard?")`
	`  Dim FileName As String`
	`  Dim Choice As String = Console.ReadLine().ToLower()`
	`  If Choice = "easy" Then`
	`  FileName = "Easy.txt"`
	`  ElseIf Choice = "medium" Then`
	`  FileName = "Medium.txt"`
		4

Question	Answer	Marks

1(b)	Else
	FileName = "Hard.txt"
	End If
	ReadWords(FileName)
	End Sub
	
	Python
	WordArray = []
	NumberWords = 0
	Choice = input("Easy, medium or hard? ").lower()
	if Choice == "easy":
	File = "Easy.txt"
	elif Choice == "medium":
	File = "Medium.txt"
	else:
	File = "Hard.txt"
	ReadWords(File)	

Question	Answer	Marks

1(c)(i)	1 mark each
	
	• Procedure start (and end) outputting the main word **and** the number of answers. **No parameters**
	• Takes word as input and compares input to **each** answer in array but **not** the main word (index 0) …
	• … replaces correct answer with a null value (e.g. "", `null`)
	… outputs if the word input is found and if the word input is not found
	• … counts the number of answers found
	• Loops MPs 2–5 until user requests to stop (enters "no")
	
	e.g.
	Java
	`public static void Play(){`
	`  System.out.println(NumberWords);`
	`  Scanner scanner = new Scanner(System.in);`
	`  String WordChosen = WordArray[0];`
	`  System.out.println("The word is " + WordChosen);`
	`  System.out.println("There are " + NumberWords + " words that can be made with 3 or more letters");`
	`  WordArray[0] = "";`
	`  Integer QuantityFound = 0;`
	`  String WordInput;`
	`  Boolean Found = false;`
	`  String Answer = "yes";`
	`  while(!(Answer.equals("no"))){`
	`  System.out.println("Enter your word or no to stop");`
	`  Answer = scanner.nextLine();`
	`  Found = false;`
	`  if(!(Answer.equals("no"))){`
	`  for(Integer x = 1; x <= NumberWords; x++){`
	`  if(Answer.equals(WordArray[x])){`
	`  WordArray[x] = "";`
	`  QuantityFound++;`
	`  System.out.println("Correct, you have found " + QuantityFound + " words");`
	`  Found = true;`
	`  }`
	`  }`
		6

Question	Answer	Marks

1(c)(i)	`if(Found == false){`
	`System.out.println("Sorry that was incorrect");`
	`}`
	`}`
	`}`
	`}`
	
	VB.NET
	`Sub Play()`
	`Dim Word As String = WordArray(0)`
	`Console.WriteLine("The word is: " & Word)`
	`Console.WriteLine("There are " & NumberWords & " words that can be made with 3 or more letters")`
	`WordArray(0) = ""`
	`Dim Contin As Boolean = True`
	`Dim QuantityFound As Integer = 0`
	`Dim Found As Boolean`
	`Dim Answer As String = "yes"`
	`While Answer <> "no"`
	`Console.WriteLine("Enter your word or no to stop")`
	`Answer = Console.ReadLine().ToLower()`
	`Found = False`
	`If Answer <> "no" Then`
	`For x = 1 To NumberWords`
	`If Answer = WordArray(x) Then`
	`WordArray(x) = ""`
	`QuantityFound = QuantityFound + 1`
	`Console.WriteLine("Correct, you have found " & QuantityFound & " words")`
	`Found = True`
	`x = NumberWords + 1`
	`End If`
	`Next x`
	`If Found = False Then`
	`Console.WriteLine("Sorry that was incorrect")`
	`End If`	

Question	Answer	Marks

1(c)(i)	End If
	End While
	End Sub
	
	Python
	def Play():
	global WordArray
	global NumberWords
	Word = WordArray[0]
	print("The word is: ", Word)
	print("There are", NumberWords-1,"words that can be made with 3 or more letters")
	WordArray[0] = ""
	Answer = "yes"
	QuantityFound = 0
	while Answer != "no":
	Answer = input("Enter your word or no to stop ").lower()
	Found = False
	if Answer != "no":
	for x in range(1, NumberWords+1):
	if Answer == WordArray[x]:
	
	WordArray[x] = ""
	QuantityFound = QuantityFound + 1
	print("Correct, you have found", QuantityFound, "words")
	Found = True
	if Found == False:
	print("Sorry that was incorrect")	

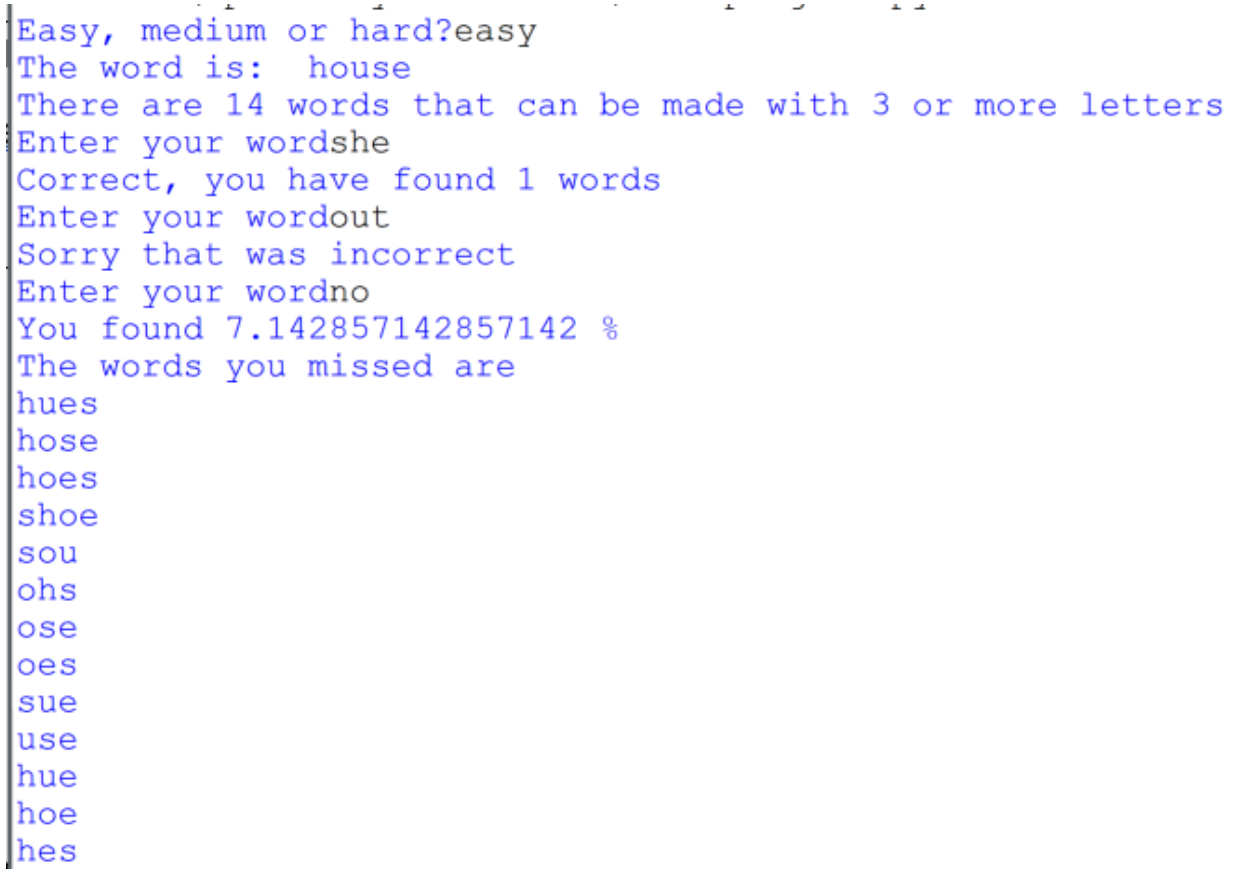
Question	Answer	Marks

1(c)(ii)	1 mark each
	
	• Calculates and outputs percentage of answers found
	• Method of identifying answers not found …
	• …and outputting those answers
	
	e.g.
	Java
	`double Correct = ((Double.valueOf(QuantityFound) / Double.valueOf(NumberWords)) * 100.0);`
	
	`System.out.println("You found " + Correct + "%");`
	`if(Correct < 100){`
	`  System.out.println("The words you missed are");`
	`  for(Integer x = 1; x <= NumberWords; x++){`
	`  if(WordArray[x] != ""){`
	`  System.out.println(WordArray[x]);`
	`  }`
	`  }`
	`}`
	
	VB.NET
	`Dim Correct As Double`
	`Correct = (QuantityFound / NumberWords) * 100`
	`Console.WriteLine("You found " & Correct & "%")`
	`If Correct < 100 Then`
	`  Console.WriteLine("The words you missed are ")`
	`  For x = 1 To NumberWords`
	`  If WordArray(x) <> "" Then`
	`  Console.WriteLine(WordArray(x))`
	`  End If`
	`  Next x`
	`End If`
		3

Question	Answer	Marks

1(c)(ii)	Python
	`Correct = (QuantityFound / (NumberWords-1)) * 100`
	`print("You found", Correct,"%")`
	`if Correct < 100:`
	`  print("The words you missed are")`
	`  for x in range(1, NumberWords+1):`
	`  if WordArray[x] != "":`
	`  print(WordArray[x])`	

1(d)(i)	1 mark for:
	• Calling `Play()`
	
	e.g.
	Java
	`Play();`
	
	VB.NET
	`Play()`
	
	Python
	`Play()`
		1

Question	Answer	Marks
1(d)(ii)	1 mark for screenshot showing the inputs "easy", "she", "out", "no" and the matching outputs e.g. 	1

Question	Answer	Marks
1(d)(iii)	1 mark for screenshot showing the inputs "hard", "fine", "fined", "idea", "no" and the matching outputs e.g. 	**1**

Question	Answer	Marks

2(a)(i)	1 mark each to max 4
	
	• Class `Node` declaration (and end where appropriate)
	• `LeftPointer`, `Data` and `RightPointer` declared as (public) integer
	• Constructor header (and end) taking **one** parameter within class …
	• …assigning parameter to `Data`, initialising `LeftPointer` and `RightPointer` to `-1`
	
	e.g.
	Java
	`public class Node{`
	`  private Integer LeftPointer;`
	`  private Integer Data;`
	`  private Integer RightPointer;`
	
	`  public Node(Integer PData){`
	`  LeftPointer = -1;`
	`  Data = PData;`
	`  RightPointer = -1;`
	`  }`
	`}`
	
	VB.NET
	`Class Node`
	`  Private LeftPointer As Integer`
	`  Private Data As Integer`
	`  Private RightPointer As Integer`
	`  Sub New(PData)`
	`  LeftPointer = -1`
	`  Data = PData`
	`  RightPointer = -1`
	`  End Sub`
	`End Class`
		4

Question	Answer	Marks

2(a)(i)	Python
	`class Node():`
	`  def __init__(self, PData):`
	`  self.__LeftPointer = -1 #int`
	`  self.__Data = PData #int`
	`  self.__RightPointer = -1 #int`	

Question	Answer	Marks

2(a)(ii)	1 mark each
	
	• 1 get method with no parameter …
	• …returning correct attribute
	• Remaining 2 correct
	
	e.g.
	Java
	`public Integer GetLeft(){`
	`  return LeftPointer;`
	`}`
	`public Integer GetRight(){`
	`  return RightPointer;`
	`}`
	`public Integer GetData(){`
	`  return Data;`
	`}`
	
	VB.NET
	`Function GetLeft()`
	`  Return LeftPointer`
	`End Function`
	`Function GetRight()`
	`  Return RightPointer`
	`End Function`
	`Function GetData()`
	`  Return Data`
	`End Function`
	
	Python
	`def GetLeft(self):`
	`  return self.__LeftPointer`
	`def GetRight(self):`
	`  return self.__RightPointer`
	`def GetData(self):`
	`  return self.__Data`
		3

Question	Answer	Marks

2(a)(iii)	1 mark each
	
	• 1 set method header (and end) with parameter …
	• …assigning parameter to correct attribute
	• Remaining 2 correct
	
	e.g.
	Java
	`public void SetLeft(Integer NewLeft){`
	`  LeftPointer = NewLeft;`
	`}`
	`public void SetRight(Integer NewRight){`
	`  RightPointer = NewRight;`
	`}`
	`public void SetData(Integer NewData){`
	`  Data = NewData;`
	`}`
	
	VB.NET
	`Sub SetLeft(NewLeft)`
	`  LeftPointer = NewLeft`
	`End Sub`
	`Sub SetRight(NewRight)`
	`  RightPointer = NewRight`
	`End Sub`
	`Sub SetData(NewData)`
	`  Data = NewData`
	`End Sub`
	
	Python
	`def SetLeft(self, NewLeft):`
	`  self.__LeftPointer = NewLeft`
	`def SetRight(self, NewRight):`
	`  self.__RightPointer = NewRight`
	`def SetData(self, NewData):`
	`  self.__Data = NewData`
		3

Question	Answer	Marks

2(b)(i)	1 mark each
	
	• Class `TreeClass` header (and end)
	• Declaration of the attributes: array `Tree` of type `Node` with **20** elements, `FirstNode` and `NumberNodes` as integers
	• Constructor header (and end) with **0** parameters, assigns `-1` to `FirstNode` and `0` to `NumberNodes` …
	• … initialises all `Tree` (20) elements to `Node` object with data value `-1`
	
	e.g.
	Java
	
	`class TreeClass{`
	`  private static Node[] Tree = new Node[20];`
	`  private static Integer FirstNode;`
	`  private static Integer NumberNodes;`
	`  public TreeClass(){`
	`  FirstNode = -1;`
	`  NumberNodes = 0;`
	`  Integer MinusOne = -1;`
	`  for(Integer x = 0; x < 20; x++){`
	`  Tree[x] = new Node(MinusOne);`
	`  }`
	`  }`
	`}`
	
	VB.NET
	`Class TreeClass`
	`  Private Tree(20) As Node`
	`  Private FirstNode As Integer`
	`  Private NumberNodes As Integer`
	
	`  Sub New()`
	`  FirstNode = -1`
	`  NumberNodes = 0`
	`  For x = 0 To 19`
	`  Tree(x) = New Node(-1)`
		4

Question	Answer	Marks

2(b)(i)	Next
	End Sub
	End Class
	
	Python
	class TreeClass():
	
	def __init__(self):
	self.__Tree = [] #type node 20 spaces
	self.__FirstNode = -1 #int
	self.__NumberNodes = 0 #int
	for x in range(20):
	self.__Tree.append(Node(-1))	

Question	Answer	Marks

2(b)(ii)	1 mark each to max 6
	
	• Method header and end, taking a `Node` as parameter
	• Checking if empty
	and (if it is empty) inserting parameter node into first position
	and updating `FirstNode` to 0
	and incrementing `NumberNodes`
	• (otherwise) inserting parameter node in array `Tree` at index `NumberNodes`
	• Accessing tree root node and comparing **data**…
	• …checking whether to go left or right …
	• … repeatedly until correct position found …
	• … updating left or right pointer for parent node **and** updating `NumberNodes`
	
	e.g.
	Java
	`public void InsertNode(Node NewNode){`
	`  Integer NodeAccess;`
	`  Integer Previous = -1;`
	`  String Direction;`
	
	`  if(NumberNodes == 0){`
	`  Tree[0] = NewNode;`
	`  FirstNode = 0;`
	`  NumberNodes++;`
	`  }else{`
	`  Tree[NumberNodes] = NewNode;`
	`  NodeAccess = FirstNode;`
	`  Direction = "";`
	`  while(NodeAccess != -1){`
	`  Previous = NodeAccess;`
	`  if(NewNode.GetData() < Tree[NodeAccess].GetData()){`
	`  NodeAccess = Tree[NodeAccess].GetLeft();`
	`  Direction = "left";`
	`  }else if(NewNode.GetData() > Tree[NodeAccess].GetData()){`
		6

Question	Answer	Marks
2(b)(ii)		

```
                NodeAccess = Tree[NodeAccess].GetRight();
                Direction = "right";
            }
        }
        if(Direction.equals("left")){
            Tree[Previous].SetLeft(NumberNodes);
        }else{
            Tree[Previous].SetRight(NumberNodes);
        }
        NumberNodes++;
    }
}
```

VB.NET

```
Sub InsertNode(NewNode)
    Dim NodeAccess As Integer
    Dim Direction As String
    Dim Previous As Integer
    If NumberNodes = 0 Then
        Tree(0) = NewNode
        FirstNode = 0
        NumberNodes += 1
    Else
        Tree(NumberNodes) = NewNode
        NodeAccess = FirstNode
        Direction = ""

        While NodeAccess <> -1
            Previous = NodeAccess
            If NewNode.GetData() < Tree(NodeAccess).GetData() Then
                NodeAccess = Tree(NodeAccess).GetLeft()
                Direction = "left"
            ElseIf NewNode.GetData() > Tree(NodeAccess).GetData() Then
                NodeAccess = Tree(NodeAccess).GetRight()
                Direction = "right"
            End If
        End While
```

Question	Answer	Marks

2(b)(ii)	(pseudocode)
	`If Direction = "left" Then`
	`Tree(Previous).SetLeft(NumberNodes)`
	`Else`
	`Tree(Previous).SetRight(NumberNodes)`
	`End If`
	`NumberNodes += 1`
	`End If`
	`End Sub`
	
	Python
	`def InsertNode(self, NewNode):`
	`if(self.__NumberNodes == 0):`
	`self.__Tree[0] = NewNode`
	`self.__FirstNode = 0`
	`self.__NumberNodes = self.__NumberNodes + 1`
	`else:`
	`self.__Tree[self.__NumberNodes] = NewNode`
	`NodeAccess = self.__FirstNode`
	`Direction = ""`
	
	`while(NodeAccess != -1):`
	`Previous = NodeAccess`
	`if NewNode.GetData() < self.__Tree[NodeAccess].GetData():`
	`NodeAccess = self.__Tree[NodeAccess].GetLeft()`
	`Direction = "left"`
	`elif NewNode.GetData() > self.__Tree[NodeAccess].GetData():`
	
	`NodeAccess = self.__Tree[NodeAccess].GetRight()`
	`Direction = "right"`
	`if(Direction == "left"):`
	
	`self.__Tree[Previous].SetLeft(self.__NumberNodes)`
	`else:`
	`self.__Tree[Previous].SetRight(self.__NumberNodes)`
	`self.__NumberNodes = self.__NumberNodes + 1`	

Question	Answer	Marks

2(b)(iii)	1 mark each
	
	• Method header (and end) with no parameter **and** if no nodes output "No nodes"
	• (otherwise) Loop from index 0 to `NumberNodes` (or equivalent)…
	• …outputting `LeftPointer`, `Data` then `RightPointer`
	• …using get methods
	
	e.g.
	Java
	`public void OutputTree(){`
	`  if(NumberNodes == 0){`
	`  System.out.println("No nodes");`
	`  }else{`
	`  for(Integer x = 0; x < NumberNodes; x++){`
	`  System.out.println(Tree[x].GetLeft() + "  " + Tree[x].GetData() + "  " + Tree[x].GetRight());`
	`  }`
	`  }`
	`}`
	
	VB.NET
	`Sub OutputTree()`
	`  If NumberNodes = 0 Then`
	`  Console.WriteLine("No nodes")`
	`  Else`
	`  For x = 0 To NumberNodes - 1`
	`  Console.WriteLine(Tree(x).GetLeft() & " " & Tree(x).GetData() & " " & Tree(x).GetRight())`
	`  Next`
	`  End If`
	`End Sub`
		4

Question	Answer	Marks

2(b)(iii)	Python
	`def OutputTree(self):`
	`  if self.__NumberNodes == 0:`
	`  print("No nodes")`
	`  else:`
	`  for x in range(0, self.__NumberNodes):`
	`  print(self.__Tree[x].GetLeft(), "  ", self.__Tree[x].GetData(), "`
	`",self.__Tree[x].GetRight())`	

2(c)(i)	1 mark for
	
	• Instance of `TreeClass` created with identifier `TheTree`
	
	Java
	`public static void main(String args[]){`
	`  TreeClass TheTree = new TreeClass();`
	`}`
	
	VB.NET
	`Sub Main(args As String())`
	`  Dim TheTree As TreeClass = New TreeClass()`
	`End Sub`
	
	Python
	`TheTree = TreeClass()`
		1

Question	Answer	Marks

2(c)(ii)	1 mark each
	
	• Creating one instance of `Node` with one correct value (e.g. 10)
	• Calling `InsertNode` for `TheTree` for **seven** new `Node`
	• All seven nodes created and inserted in order
	• Calling `OutputTree()` for the tree created
	
	Java
	
	`TreeClass TheTree = new TreeClass();`
	`TheTree.InsertNode(new Node(10));`
	`TheTree.InsertNode(new Node(11));`
	`TheTree.InsertNode(new Node(5));`
	`TheTree.InsertNode(new Node(1));`
	`TheTree.InsertNode(new Node(20));`
	`TheTree.InsertNode(new Node(7));`
	`TheTree.InsertNode(new Node(15));`
	`TheTree.OutputTree();`
	
	VB.NET
	
	`Dim TheTree As TreeClass = New TreeClass()`
	`TheTree.InsertNode(New Node(10))`
	`TheTree.InsertNode(New Node(11))`
	`TheTree.InsertNode(New Node(5))`
	`TheTree.InsertNode(New Node(1))`
	`TheTree.InsertNode(New Node(20))`
	`TheTree.InsertNode(New Node(7))`
	`TheTree.InsertNode(New Node(15))`
	`TheTree.OutputTree()`
		4

Question	Answer	Marks

2(c)(ii)	Python
	`TheTree = TreeClass()`
	`TheTree.InsertNode(Node(10))`
	`TheTree.InsertNode(Node(11))`
	`TheTree.InsertNode(Node(5))`
	`TheTree.InsertNode(Node(1))`
	`TheTree.InsertNode(Node(20))`
	`TheTree.InsertNode(Node(7))`
	`TheTree.InsertNode(Node(15))`
	`TheTree.OutputTree()`	

2(c)(iii)	1 mark for correct output e.g. 2 10 1 -1 11 4 3 5 5 -1 1 -1 6 20 -1 -1 7 -1 -1 15 -1	**1**

Question	Answer	Marks

3(a)	1 mark for
	
	• `NumberArray` declared (in main) with the 7 correct integer values in the correct order
	
	e.g.
	Java
	`Integer[] NumberArray = new Integer[7];`
	`NumberArray[0] = 100;`
	`NumberArray[1] = 85;`
	`NumberArray[2] = 644;`
	`NumberArray[3] = 22;`
	`NumberArray[4] = 15;`
	`NumberArray[5] = 8;`
	`NumberArray[6] = 1;`
	
	VB.NET
	`Dim NumberArray(7) As Integer`
	`NumberArray(0) = 100`
	`NumberArray(1) = 85`
	`NumberArray(2) = 644`
	`NumberArray(3) = 22`
	`NumberArray(4) = 15`
	`NumberArray(5) = 8`
	`NumberArray(6) = 1`
	`EndSub`
	
	Python
	`NumberArray = [100, 85, 644, 22, 15, 8, 1]`
		1

Question	Answer	Marks

3(b)(i)	1 mark each
	
	• Recursive function header (and end where appropriate) taking only 2 parameters with a recursive call
	• Correct base case and return
	• Correct while loop control and the loop
	• All correct and structure followed
	
	e.g.
	Java
	`public static Integer[] RecursiveInsertion(Integer[] IntegerArray, Integer NumberElements){`
	`  Integer LastItem;`
	`  Integer CheckItem;`
	`  if(NumberElements <= 1){`
	`  return IntegerArray;`
	`  }else{`
	`  RecursiveInsertion(IntegerArray, NumberElements - 1);`
	`  LastItem = IntegerArray[NumberElements - 1];`
	`  CheckItem = NumberElements - 2;`
	`  }`
	`  Boolean LoopAgain = true;`
	`  if(CheckItem < 0){`
	`  LoopAgain = false;`
	`  }else if(IntegerArray[CheckItem] < LastItem){`
	`  LoopAgain = false;`
	`  }`
	`  while(LoopAgain){`
	`  IntegerArray[CheckItem + 1] = IntegerArray[CheckItem];`
	`  CheckItem = CheckItem - 1;`
	`  if(CheckItem < 0){`
	`  LoopAgain = false;`
	`  }else if(IntegerArray[CheckItem] < LastItem){`
	`  LoopAgain = false;`
	`  }`
	`  }`
		4

Question	Answer	Marks

3(b)(i)	`    IntegerArray[CheckItem + 1] = LastItem;`
	`    return IntegerArray;`
	`}`
	
	VB.NET
	`Function RecursiveInsertion(IntegerArray, NumberElements)`
	`    Dim LastItem, CheckItem As Integer`
	`    If NumberElements <= 1 Then`
	`        Return IntegerArray`
	`    Else`
	`        RecursiveInsertion(IntegerArray, NumberElements - 1)`
	`        LastItem = IntegerArray(NumberElements - 1)`
	`        CheckItem = NumberElements - 2`
	`    End If`
	
	`    Dim LoopAgain As Boolean = True`
	`    If CheckItem < 0 Then`
	`        LoopAgain = False`
	`    ElseIf IntegerArray(CheckItem) < LastItem Then`
	`        LoopAgain = False`
	`    End If`
	`    While LoopAgain`
	`        IntegerArray(CheckItem + 1) = IntegerArray(CheckItem)`
	`        CheckItem = CheckItem - 1`
	`        If CheckItem < 0 Then`
	`            LoopAgain = False`
	`        ElseIf IntegerArray(CheckItem) < LastItem Then`
	`            LoopAgain = False`
	`        End If`
	`    End While`
	`    IntegerArray(CheckItem + 1) = LastItem`
	`    Return IntegerArray`
	`End Function`	

Question	Answer	Marks

3(b)(i)	Python
	`def RecursiveInsertion(IntegerArray, NumberElements):`
	
	`    if NumberElements <= 1:`
	`        return IntegerArray`
	
	`    RecursiveInsertion(IntegerArray,NumberElements - 1)`
	`    LastItem = IntegerArray[NumberElements - 1]`
	`    CheckItem = NumberElements - 2`
	
	`    LoopAgain = True`
	`    if CheckItem < 0:`
	`        LoopAgain = False`
	`    elif IntegerArray[CheckItem] < LastItem:`
	`        LoopAgain = False`
	
	`    while (LoopAgain):`
	`        IntegerArray[CheckItem + 1] = IntegerArray[CheckItem]`
	`        CheckItem = CheckItem - 1`
	`        if CheckItem < 0:`
	`            LoopAgain = False`
	`        elif IntegerArray[CheckItem] < LastItem:`
	`            LoopAgain = False`
	
	`    IntegerArray[CheckItem + 1] = LastItem`
	`    return IntegerArray`	

Question	Answer	Marks

3(b)(ii)	1 mark each
	
	• Calling `RecursiveInsertion()` with array and number of elements as parameters
	• Outputting "recursive" and then each element in **returned array**
	
	e.g.
	Java
	`Integer[] SortedArray = new Integer[7];`
	`SortedArray = RecursiveInsertion(NumberArray, 7);`
	`System.out.println("Recursive");`
	`for(Integer x = 0; x < 7; x++){`
	`  System.out.println(SortedArray[x]);`
	`}`
	
	VB.NET
	`SortedArray = RecursiveInsertion(NumberArray, 7)`
	`Console.WriteLine("Recursive")`
	`For x = 0 To 6`
	`  Console.WriteLine(SortedArray(x))`
	`Next x`
	
	Python
	
	`SortedArray = RecursiveInsertion(NumberArray, len(NumberArray))`
	`print("Recursive", SortedArray)`
		2

3(b)(iii)	1 mark for screenshot with:
	
	`Recursive`
	`1`
	`8`
	`15`
	`22`
	`85`
	`100`
	`644`
		1

Question	Answer	Marks

3(c)(i)	1 mark each
	
	• Iterative insertion algorithm header (and end)
	and taking array parameter (minimum)
	and returning the sorted array
	• External loop while there are still elements left (e.g. `NumberElements > 0`)…
	• … internal loop …
	• … and selection accurate
	
	e.g.
	Java
	`public static Integer[] IterativeInsertion(Integer[] IntegerArray, Integer NumberElements){`
	`  Integer LastItem;`
	`  Integer CheckItem;`
	`  while(NumberElements > 0){`
	`  LastItem = IntegerArray[NumberElements - 1];`
	`  CheckItem = NumberElements - 2;`
	`  Boolean LoopAgain = true;`
	`  if(CheckItem < 0){`
	`  LoopAgain = false;`
	`  }else if(IntegerArray[CheckItem] < LastItem){`
	`  LoopAgain = false;`
	`  }`
	`  while(LoopAgain){`
	`  IntegerArray[CheckItem + 1] = IntegerArray[CheckItem];`
	`  CheckItem = CheckItem - 1;`
	`  if(CheckItem < 0){`
	`  LoopAgain = false;`
	`  }else if(IntegerArray[CheckItem] < LastItem){`
	`  LoopAgain = false;`
	`  }`
	`  }`
	`  IntegerArray[CheckItem + 1] = LastItem;`
	`  NumberElements = NumberElements - 1;`
	`  }`
		4

Question	Answer	Marks

3(c)(i)	`    return IntegerArray;`
	`}`
	
	VB.NET
	`Function IterativeInsertion(IntegerArray, NumberElements)`
	`    Dim LastItem, CheckItem As Integer`
	`    While NumberElements > 0`
	`        LastItem = IntegerArray(NumberElements - 1)`
	`        CheckItem = NumberElements - 2`
	`        Dim LoopAgain As Boolean = True`
	`        If CheckItem < 0 Then`
	`            LoopAgain = False`
	`        ElseIf IntegerArray(CheckItem) < LastItem Then`
	`            LoopAgain = False`
	`        End If`
	`        While LoopAgain`
	`            IntegerArray(CheckItem + 1) = IntegerArray(CheckItem)`
	`            CheckItem = CheckItem - 1`
	
	`            If CheckItem < 0 Then`
	`                LoopAgain = False`
	`            ElseIf IntegerArray(CheckItem) <= LastItem Then`
	`                LoopAgain = False`
	`            End If`
	`        End While`
	`        IntegerArray(CheckItem + 1) = LastItem`
	`        NumberElements = NumberElements - 1`
	
	`    End While`
	`    Return IntegerArray`
	`End Function`	

Question	Answer	Marks

3(c)(i)	Python
	`def IterativeInsertion(IntegerArray, NumberElements):`
	`  while NumberElements > 0:`
	`  LastItem = IntegerArray[NumberElements - 1]`
	`  CheckItem = NumberElements - 2`
	`  LoopAgain = True`
	`  if CheckItem < 0:`
	`  LoopAgain = False`
	`  elif IntegerArray[CheckItem] < LastItem:`
	`  LoopAgain = False`
	`  while(LoopAgain):`
	`  IntegerArray[CheckItem + 1] = IntegerArray[CheckItem]`
	`  CheckItem = CheckItem - 1`
	`  if CheckItem < 0:`
	`  LoopAgain = False`
	`  elif IntegerArray[CheckItem] <= LastItem:`
	`  LoopAgain = False`
	
	`  IntegerArray[CheckItem + 1] = LastItem`
	`  NumberElements = NumberElements - 1`
	`  return IntegerArray`	

Question	Answer	Marks

3(c)(ii)	1 mark each
	
	• Calling `IterativeInsertion()` with original unsorted array
	and outputting "Iterative"
	and outputting the content of the **returned** array
	
	e.g.
	Java
	`Integer[] Sorted2Array = new Integer[7];`
	`Sorted2Array = IterativeInsertion(NumberArray, 7);`
	`System.out.println("Iterative");`
	`for(Integer x = 0; x < 7; x++){`
	`  System.out.println(Sorted2Array[x]);`
	`}`
	
	VB.NET
	`Sorted2Array = IterativeInsertion(NumberArray, 7)`
	`Console.WriteLine("Iterative")`
	`For x = 0 To 6`
	`  Console.WriteLine(Sorted2Array(x))`
	`Next x`
	
	Python
	`Sorted2Array = IterativeInsertion(NumberArray, len(NumberArray))`
	`print("Iterative", Sorted2Array)`
		1

3(c)(iii)	1 mark for screenshot showing: Iterative 1 8 15 22 85 100 644	**1**

Question	Answer	Marks

3(d)(i)	1 mark each to max 6
	
	• Recursive function `BinarySearch` taking 4 parameters
	• Suitable base case (e.g. `First > Last`) …
	• … returning `-1`
	• Calculating integer middle element
	• Comparing `ToFind` with `Middle` element **and** returning `Middle` if equal
	• If `ToFind` **<** `Middle`, recursive call with `Last` as `Middle - 1`
	• If `ToFind` **>** `Middle`, recursive call with `First` as `Middle + 1`
	
	Java
	`public static Integer BinarySearch(Integer[] IntegerArray, Integer First, Integer Last, Integer ToFind){`
	`  Integer Middle;`
	`  if(First > Last){;`
	`  return -1;`
	`  }else{`
	`  Middle = (Last + First) / 2;`
	
	`  if(IntegerArray[Middle].equals(ToFind)){`
	`  return Middle;`
	`  }else if(IntegerArray[Middle] > ToFind){`
	`  return BinarySearch(IntegerArray, First, Middle - 1, ToFind);`
	`  }else{`
	`  return BinarySearch(IntegerArray, Middle + 1, Last, ToFind);`
	`  }`
	`  }`
	`}`
		6

Question	Answer	Marks

3(d)(i)	VB.NET
	`Function BinarySearch(IntegerArray, First, Last, ToFind)`
	`  Dim Middle As Integer`
	`  If First > Last Then`
	`  Return -1`
	`  Else`
	`  Middle = (Last + First) \ 2`
	`  If IntegerArray(Middle) = ToFind Then`
	
	`  Return Middle`
	`  ElseIf IntegerArray(Middle) > ToFind Then`
	`  Return BinarySearch(IntegerArray, First, Middle - 1, ToFind)`
	`  Else`
	`  Return BinarySearch(IntegerArray, Middle + 1, Last, ToFind)`
	`  End If`
	`  End If`
	`End Function`
	
	Python
	`def BinarySearch(IntegerArray, First, Last, ToFind):`
	`  if First > Last:`
	`  return -1`
	`  else:`
	`  Middle = int((Last + First) / 2)`
	`  if IntegerArray[Middle] == ToFind:`
	`  return Middle`
	`  elif IntegerArray[Middle] > ToFind:`
	`  return BinarySearch(IntegerArray, First, Middle - 1, ToFind)`
	`  else:`
	`  return BinarySearch(IntegerArray, Middle + 1, Last, ToFind)`	

<table>
<tr><th>Question</th><th>Answer</th><th>Marks</th></tr>
<tr><td>3(d)(ii)</td><td>1 mark each

• Calling <code>BinarySearch</code> function with sorted array, <code>0</code>, <code>6(/len(array)-1)</code>, <code>644</code> as parameters
• Checking return value and outputting <code>"Not found"</code> if <code>-1</code> and returned index otherwise

Java
<pre>Position = BinarySearch(Sorted2Array, 0, 6, 644);
if(Position == -1){
   System.out.println("Not found");
}else{
   System.out.println(Position);
}</pre>
VB.NET
<pre>Position = BinarySearch(Sorted2Array, 0, 6, 644)
If Position = -1 Then
   Console.WriteLine("Not found")
Else
   Console.WriteLine(Position)
End If</pre>
Python
<pre>Position = BinarySearch(Sorted2Array, 0, len(NumberArray)-1, 644)
if Position == -1:
    print("Not found")
else:
    print(Position)</pre></td><td>2</td></tr>
<tr><td>3(d)(iii)</td><td>1 mark for screenshot showing found in index 6
e.g.

<pre>Recursive [1, 8, 15, 22, 85, 100, 644]
Iterative [1, 8, 15, 22, 85, 100, 644]
6</pre></td><td>1</td></tr>
</table>