CAMBRIDGE
International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/41**

Paper 4 Practical **May/June 2025**

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 2025 series for most Cambridge IGCSE, Cambridge International A and AS Level components, and some Cambridge O Level components.

This document consists of **43** printed pages.

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.

Annotations guidance for centres

Examiners use a system of annotations as a shorthand for communicating their marking decisions to one another. Examiners are trained during the standardisation process on how and when to use annotations. The purpose of annotations is to inform the standardisation and monitoring processes and guide the supervising examiners when they are checking the work of examiners within their team. The meaning of annotations and how they are used is specific to each component and is understood by all examiners who mark the component.

We publish annotations in our mark schemes to help centres understand the annotations they may see on copies of scripts. Note that there may not be a direct correlation between the number of annotations on a script and the mark awarded. Similarly, the use of an annotation may not be an indication of the quality of the response.

The annotations listed below were available to examiners marking this component in this series.

Annotations

Annotation	Meaning
BOD	Benefit of the doubt
λ	To indicate where a key word/phrase/code is missing
✗	Incorrect
FT	Follow through
〰	Indicate a point in an answer
Highlighted text	To draw attention to a particular aspect or to indicate where parts of an answer have been combined
I	Ignore
NAQ	Not answered question
NBOD	No benefit of doubt given
NE	No examples or not enough

Annotation	**Meaning**
⌇	Not relevant or used to separate parts of an answer
Off-page comment	Allows comments to be entered at the bottom of the RM marking window and then displayed when the associated question item is navigated to.
REP	Repetition
SEEN	Indicates that work or a page has been seen including blank answer spaces and blank pages.
✓	Correct
TV	Too vague

Mark scheme abbreviations

- **Bold** in mark scheme means that idea is required.
- / in mark scheme means alternative.
- // in mark scheme means alternative solution that gains the same mark point.
- … at the end of one mark point without a … at the start of the next just means the sentence follows on. There is no dependency.
- … at the end of one mark point and … at the start of the next, this means the second cannot be awarded without the first.
- () means what is in the brackets is not required, or it is not required in some languages but may be required in others.

Question	Answer	Marks
1(a)	1 mark each • (global) Declaration of 1D array `Queue`, 20 elements initialised with `−1` • (global) `HeadPointer` and `TailPointer` initialised to `−1`, `NumberItems` initialised with `0`	**2**

Question	Answer	Marks

Example program code:

Python

```
Queue = [-1 for x in range(20)]
HeadPointer = -1
TailPointer = -1
NumberItems = 0
```

VB.NET

```
Dim Queue(20) As Integer
Dim HeadPointer As Integer
Dim TailPointer As Integer
Dim NumberItems As Integer
For x = 0 To 19
   Queue(x) = -1
Next
HeadPointer = -1
TailPointer = -1
NumberItems = 0
```

Java

```
public static Integer[] Queue = new Integer[20];
public static Integer HeadPointer;
public static Integer TailPointer;
public static Integer NumberItems;
public static void main(String args[]){

       for(Integer X = 0; X < 20; X++){
           Queue[X] = -1;
       }
       HeadPointer = -1;
       TailPointer = -1;
       NumberItems = 0;
}
```

Question	Answer	Marks
1(b)	1 mark each • Function header (and close) taking 1 (integer) parameter and returning a Boolean value in all cases • Checking if queue is full (`NumberItems = 20`) and returning `FALSE` • Checking if queue is empty (`NumberItems = 0`) then and updating `TailPointer` and `HeadPointer` appropriately • Incrementing `TailPointer` and `NumberItems` in appropriate place … • … looping back to 0 for `TailPointer` if at end of structure • Storing parameter in `Queue[TailPointer]` (after increment) and returning `TRUE`	**6**

Question	Answer	Marks

Example program code:

Python
```
def Enqueue(InputData):
    global Queue
    global HeadPointer
    global TailPointer
    global NumberItems
    if NumberItems >= 20:
        return False

    if TailPointer <= -1:
        TailPointer = 0
        HeadPointer = 0
        Queue[TailPointer] = InputData
    else:
        TailPointer = TailPointer + 1
        if TailPointer == 20:
            TailPointer = 0

        Queue[TailPointer] = InputData
    NumberItems +=1
    return True
```

VB.NET
```
Function Enqueue(InputData)
   If NumberItems >= 20 Then
      Return False
   End If

   If TailPointer <= -1 Then
      TailPointer = 0
      HeadPointer = 0
      Queue(TailPointer) = InputData
```

Question	Answer	Marks

```
   Else
      TailPointer = TailPointer + 1
      If TailPointer = 20 Then
          TailPointer = 0
      End If
      Queue(TailPointer) = InputData
   End If
   NumberItems = NumberItems + 1
   Return True
End Function
```

Java

```
public static Boolean Enqueue(Integer InputData){
       if(NumberItems >= 20){
           return false;
       }
       if(TailPointer <= -1){
           TailPointer = 0;
           HeadPointer = 0;
           Queue[TailPointer] = InputData;
       }else{
                TailPointer++;
           if(TailPointer == 20){
                TailPointer = 0;
           }
           Queue[TailPointer] = InputData;
       }
       NumberItems++;
       return true;
}
```

Question	Answer	Marks
1(c)	1 mark each • Calling `Enqueue()` with 1 to 25 (inclusive) in order • … storing/using return value in selection …outputting `Successful` with integer **and** outputting `Unsuccessful` with integer correctly	3

Example program code:

Python

```
for X in range(1, 26):
    ReturnValue = Enqueue(X)
    if ReturnValue == True:
        print(x,"Successful")
    else:
        print(x,"Unsuccessful")
```

VB.NET

```
Dim ReturnValue As Boolean
For x = 1 To 25
   ReturnValue = Enqueue(x)
   If ReturnValue = True Then
      Console.WriteLine(x & "Successful " )
   Else
      Console.WriteLine(x & "Unsuccessful ")
   End If
Next x
```

Java

```
Boolean ReturnValue;
for(Integer X = 1; X < 26; X++){
       ReturnValue = Enqueue(X);
       if(ReturnValue == true){
           System.out.println(X + "Successful ");
       }else{
           System.out.println(X + "Unsuccessful ");
       }
}
```

Question	Answer	Marks
1(d)	1 mark each • `Dequeue()` header (and close) and checking if queue is empty (`NumberItems = 0`) and returning `-1` • Returning data at `HeadPointer` • Incrementing `HeadPointer …` • … and catching if = 20 to return to 0 • Decrementing `NumberItems` • Resetting `HeadPointer` **and** `TailPointer` when queue is empty	**6**

Question	Answer	Marks

Example program code:

Python

```
def Dequeue():
    global Queue
    global HeadPointer
    global TailPointer
    global NumberItems
    if NumberItems <= 0:
        return -1
    else:
        ReturnValue = Queue[HeadPointer]
        HeadPointer +=1
        if HeadPointer >= 20:
            HeadPointer = 0
        NumberItems -=1
        if NumberItems == 0:
            HeadPointer = -1
            TailPointer = -1
        return ReturnValue
```

VB.NET

```
Function Dequeue()
   Dim ReturnValue As Integer
   If NumberItems <= 0 Then
       Return -1
   Else
      ReturnValue = Queue(HeadPointer)
      HeadPointer = HeadPointer + 1
      If HeadPointer >= 20 Then
         HeadPointer = 0
      End If
      NumberItems = NumberItems – 1
      If NumberItems = 0 Then
         HeadPointer = -1
         TailPointer = -1
```

Question	Answer	Marks

```
      End If
      Return ReturnValue
   End If
End Function
```

Java

```
public static Integer Dequeue(){
       Integer ReturnValue;
       if(NumberItems <= 0){
           return -1;
       }else{
           ReturnValue = Queue[HeadPointer];
           HeadPointer++;
           if(HeadPointer >= 20){
                HeadPointer = 0;
           }
           NumberItems--;
           if(NumberItems == 0){
                HeadPointer = -1;
                TailPointer = -1;
           }
           return ReturnValue;
       }
}
```

Question	Answer	Marks
1(e)(i)	1 mark each • Calling `Dequeue()` twice • … outputting return value from both calls	**2**

Example program code:
Python

```
NextValue = Dequeue()
print(NextValue)
NextValue = Dequeue()
print(NextValue)
```

VB.NET

```
Dim NextValue As Integer
NextValue = Dequeue()
Console.WriteLine(NextValue)
NextValue = Dequeue()
Console.WriteLine(NextValue)
```

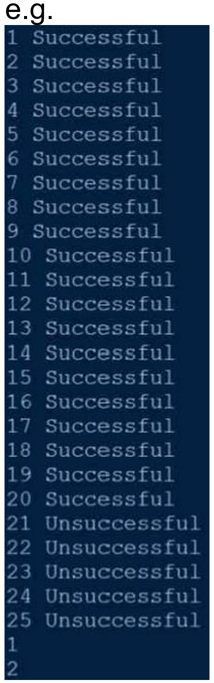
Java

```
System.out.println(Dequeue());
System.out.println(Dequeue());
```

Question	Answer	Marks
1(e)(ii)	1 mark for output showing: • 1 to 20 with Successful 21 to 25 with Unsuccessful 1 and 2 output	**1**

Question	Answer	Marks

e.g.



```
1 Successful
2 Successful
3 Successful
4 Successful
5 Successful
6 Successful
7 Successful
8 Successful
9 Successful
10 Successful
11 Successful
12 Successful
13 Successful
14 Successful
15 Successful
16 Successful
17 Successful
18 Successful
19 Successful
20 Successful
21 Unsuccessful
22 Unsuccessful
23 Unsuccessful
24 Unsuccessful
25 Unsuccessful
1
2
```

Question	Answer	Marks
2(a)	1 mark each to max 7 • Function header (and end) • Prompt to enter filename and reading input • Opening the file (to read) and closing the file in an appropriate place • Looping until EOF … • … reading each line in the file … • … (removing line break and) inserting in array • Returning populated array • Exception handling try catch with appropriate output	**7**

Question	Answer	Marks

Example program code:

Python

```
def ReadData():
    DataList = []
    FileName = input("Enter the filename")
    try:
        File = open(FileName)
        for Line in File:
            DataList.append(Line)
        File.close()
    except:
        print("Cannot open file")
    return DataList
```

VB.NET

```
Function ReadData()
   Dim DataList(100) As String
   Console.WriteLine("Enter the filename")
   Dim FileName As String = Console.ReadLine()
   NumberItems = 0

   Try
      Dim FileReader As New System.IO.StreamReader(FileName)
      While Not FileReader.EndOfStream
         DataList(NumberItems) = FileReader.ReadLine()
         NumberItems = NumberItems + 1
      End While
      FileReader.Close()
   Catch ex As Exception
      Console.WriteLine("Cannot open or read from file")
   End Try
   Return DataList
End Function
```

Question	Answer	Marks

Java

```
public static String[] ReadData(){
       String[] DataList = new String[100];
       System.out.println("Enter the filename");
       Scanner scanner = new Scanner(System.in);
       String FileName = scanner.nextLine();
       NumberItems = 0;
       try{
          FileReader f = new FileReader(FileName);
          try{
               BufferedReader Reader = new BufferedReader(f);
               String Line = Reader.readLine();
               Line = Line.replace("\n","");

               while (Line != null){
                    DataList[NumberItems] = Line;
                    NumberItems++;
                    Line = Reader.readLine();
                    if(Line != null){
                       Line = Line.replace("\n","");
                    }
               }
               Reader.close();
          }catch(IOException ex){
          }
       }catch(FileNotFoundException e){
          System.out.println("File not found");
       }
       return DataList;
}
```

Question	Answer	Marks
2(b)	1 mark each • Procedure header (and end) taking (1D array) `DataArray` (of strings) as a parameter • Declaration/use of 6 1D arrays (equivalent), one for each colour • Looping through each line in parameter `DataArray` … • … splitting by comma • Comparing 2nd value/colour to each colour to select array … • … storing 1st value/integer in correct array	**6**

Question	Answer	Marks

Example program code:

Python

```
def SplitData(DataArray):
    Red = []
    Green = []
    Blue = []
    Orange = []
    Yellow = []
    Pink = []

    for Line in DataArray:
        SplitLine = Line.split(",")

        if SplitLine[1].strip() == "red":
            Red.append(SplitLine[0])
        elif SplitLine[1].strip() == "green":
            Green.append(SplitLine[0])
        elif SplitLine[1].strip() == "blue":
            Blue.append(SplitLine[0])
        elif SplitLine[1].strip() == "orange":
            Orange.append(SplitLine[0])
        elif SplitLine[1].strip() == "yellow":
            Yellow.append(SplitLine[0])
        else:
            Pink.append(SplitLine[0])
```

VB.NET

```
Sub SplitData(DataArray())
   Dim Red(30) As String
   Dim Green(30) As String
   Dim Blue(30) As String
   Dim Orange(30) As String
   Dim Yellow(30) As String
   Dim Pink(30) As String
```

Question	Answer	Marks

```
Dim RedNumber As Integer = 0
Dim GreenNumber As Integer = 0
Dim BlueNumber As Integer = 0
Dim OrangeNumber As Integer = 0
Dim YellowNumber As Integer = 0
Dim PinkNumber As Integer = 0
Dim x As Integer = 0
Dim TempDataFromFile(1) As String
Dim DataList(100, 1) As String

For x = 0 To NumberItems - 1
   TempDataFromFile = (DataArray(x)).Split(",")
   DataList(x, 0) = TempDataFromFile(0)
   DataList(x, 1) = TempDataFromFile(1)
Next x
x = 0
While DataList(x, 0) IsNot Nothing
   If DataList(x, 1) = "red" Then
      Red(RedNumber) = DataList(x, 0)
      RedNumber = RedNumber + 1
   ElseIf DataList(x, 1) = "green" Then
      Green(GreenNumber) = DataList(x, 0)
      GreenNumber = GreenNumber + 1
   ElseIf DataList(x, 1) = "blue" Then
      Blue(BlueNumber) = DataList(x, 0)
      BlueNumber = BlueNumber + 1
   ElseIf DataList(x, 1) = "orange" Then
      Orange(OrangeNumber) = DataList(x, 0)
      OrangeNumber = OrangeNumber + 1
   ElseIf DataList(x, 1) = "yellow" Then
      Yellow(YellowNumber) = DataList(x, 0)
      YellowNumber = YellowNumber + 1
   ElseIf DataList(x, 1) = "pink" Then
      Pink(PinkNumber) = DataList(x, 0)
      PinkNumber = PinkNumber + 1
   End If
```

Question	Answer	Marks

```
      x = x + 1
   End While
End Sub
```

Java

```
public static void SplitData(String[] DataArray){
       String[] Red = new String[30];
       String[] Green = new String[30];
       String[] Blue = new String[30];
       String[] Orange = new String[30];
       String[] Yellow = new String[30];
       String[] Pink = new String[30];
       Integer RedNumber = 0;
       Integer GreenNumber = 0;
       Integer BlueNumber = 0;
       Integer OrangeNumber = 0;
       Integer YellowNumber = 0;
       Integer PinkNumber = 0;
       Integer x = 0;
       String[] TempDataFromFile;
       String[][] DataList = new String[100][2];

       for(x = 0; x < 72; x++){
           TempDataFromFile = DataArray[x].split(",");
           DataList[x][0] = TempDataFromFile[0];
           DataList[x][1] = TempDataFromFile[1];
       }

       x = 0;
       while(DataList[x][0] != null){
           if (DataList[x][1].compareTo("red") == 0) {
                RedNumber] = DataList[x][0];
                RedNumber = RedNumber + 1;
            }else if (DataList[x][1].compareTo("green") == 0) {
                Green[GreenNumber] = DataList[x][0];
                GreenNumber = GreenNumber + 1;
```

Question	Answer	Marks

```
            }else if (DataList[x][1].compareTo("blue") == 0) {
                Blue[BlueNumber] = DataList[x][0];
                BlueNumber = BlueNumber + 1;
            }else if (DataList[x][1].compareTo("orange") == 0) {
                Orange[OrangeNumber] = DataList[x][0];
                OrangeNumber = OrangeNumber + 1;
            }else if (DataList[x][1].compareTo("yellow") == 0) {
                Yellow[YellowNumber] = DataList[x][0];
                YellowNumber = YellowNumber + 1;
            }else if (DataList[x][1].compareTo("pink") == 0) {
                Pink[PinkNumber] = DataList[x][0];
                PinkNumber = PinkNumber + 1;
            }
       x = x + 1;
   }
}
```

Question	Answer	Marks
2(c)	1 mark each • Procedure header taking (1D) array and filename as parameters, opening file to append and closing file (in appropriate place) • Looping through each item in array parameter … • … writing to the file • … with new line break between each line • Using exception handling try and catch with suitable output	**5**

Example program code:

Python
```
def StoreData(DataToStore, FileName):
    try:
        File = open(FileName,"a+")
        for Item in DataToStore:
            File.write(Item)
            File.write("\n")
        File.close()
    except:
        print("Cannot create or write to file")
```

VB.NET
```
Sub StoreData(DataToStore(), FileName)
   Dim FileWriter As IO.StreamWriter = New IO.StreamWriter(FileName, False)
   Dim x As Integer = 0
   Try
      While DataToStore(x) IsNot Nothing
         FileWriter.WriteLine(DataToStore(x))
         x = x + 1
      End While
      FileWriter.Close()
   Catch ex As Exception
      Console.WriteLine("Cannot open or write to file")
   End Try
End Sub
```

Question	Answer	Marks

Java

```
public static void StoreData(String[] DataToStore, String FileName){
       File TheFile = new File(FileName);

       try{
           FileWriter FW = new FileWriter(TheFile, true);
           Integer X = 0;
           while(DataToStore[X] != null){
               FW.write(DataToStore[X]);
               X++;
               FW.write("\n");
           }
           FW.close();
       }catch(IOException ex){
           System.out.println("Cannot open or write to file");
       }
}
```

Question	Answer	Marks
2(d)	1 mark each • Calling `StoreData` with one array and filename • Calling `StoreData` with remaining 5 arrays and filename	**2**

Example program code:

Python

```
StoreData(Red, "Red.txt")
StoreData(Green, "Green.txt")
StoreData(Blue, "Blue.txt")
StoreData(Orange, "Orange.txt")
StoreData(Yellow, "Yellow.txt")
StoreData(Pink, "Pink.txt")
```

VB.NET

```
StoreData(Red, "Red.txt")
StoreData(Green, "Green.txt")
StoreData(Blue, "Blue.txt")
StoreData(Orange, "Orange.txt")
StoreData(Yellow, "Yellow.txt")
StoreData(Pink, "Pink.txt")
```

Java

```
StoreData(Red, "Red.txt");
StoreData(Green, "Green.txt");
StoreData(Blue, "Blue.txt");
StoreData(Orange, "Orange.txt");
StoreData(Yellow, "Yellow.txt");
StoreData(Pink, "Pink.txt");
```

Question	Answer	Marks
2(e)(i)	1 mark each • Calling `ReadData()` … • … and storing/using return value • Calling `SplitData()` with returned array as a parameter	**3**

Example program code:

Python

```
DataFromFile = ReadData()
SplitData(DataFromFile)
```

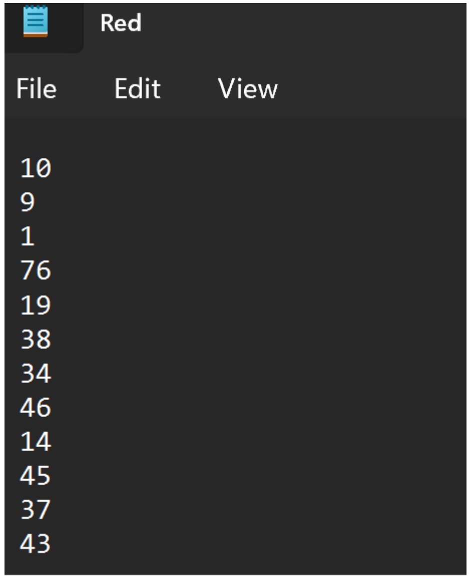
VB.NET

```
Sub Main(args As String())
   Dim DataFromFile(,) As String = ReadData()
   SplitData(DataFromFile)
End Sub
```

Java

```
public static void main(String args[]){
        String[][] DataFromFile = ReadData();
        SplitData(DataFromFile);
}
```

Question	Answer	Marks
2(e)(ii)	1 mark screenshots showing • Prompt and input of filename `TheData.txt` • Screenshot of data in red file. Filename must be shown in same screenshot as data	**2**

Question	Answer	Marks
	 10 9 1 76 19 38 34 46 14 45 37 43	

Question	Answer	Marks
3(a)(i)	1 mark each • Class header (and end where appropriate) • Constructor header (and end where appropriate) with (min) one parameter (integer) within class • 3 attributes with correct data types • `NodeData` has parameter assigned, `LeftNode` and `RightNode` are assigned null within constructor	**4**

Question	Answer	Marks

Example program code:

Python
```
class Node:
    def _init_(self, pNodeData):
        self._NodeData = pNodeData #integer
        self._LeftNode = None #node
        self._RightNode = None #node
```

VB.NET
```
Class Node
   Private NodeData As Integer
   Private LeftNode As Node
   Private RightNode As Node
   Sub New(pNodeData)
      NodeData = pNodeData
      LeftNode = Nothing
      RightNode = Nothing
   End Sub
End Class
```

Java
```
class Node{
       public Integer NodeData;
       public Node LeftNode;
       public Node RightNode;
       public Node(Integer pNodeData){
           NodeData = pNodeData;
           LeftNode = null;
           RightNode = null;
       }
}
```

Question	Answer	Marks
3(a)(ii)	1 mark each • 1 get method with no parameter … • … returning correct value • 2nd and 3rd correct get methods	**3**

Example program code:

Python

```
def GetLeft(self):
    return self._LeftNode
def GetRight(self):
    return self._RightNode
def GetData(self):
    return self._NodeData
```

VB.NET

```
Function GetLeft()
   Return LeftNode
End Function
Function GetRight()
   Return RightNode
End Function
Function GetData()
   Return NodeData
End Function
```

Java

```
public Integer GetData(){
       return NodeData;
}
public Node GetLeft(){
       return LeftNode;
}
public Node GetRight(){
       return RightNode;
}
```

Question	Answer	Marks
3(a)(iii)	1 mark each • 1 set method taking parameter of type Node … • … assigning to correct attribute • 2nd correct set method	**3**

Example program code:

Python

```
def SetLeft(self, NewNode):
    self._LeftNode = NewNode
def SetRight(self, NewNode):
    self._RightNode = NewNode
```

VB.NET

```
Sub SetLeft(NewNode)
   LeftNode = NewNode
End Sub
Sub SetRight(NewNode)
   RightNode = NewNode
End Sub
```

Java

```
public void SetLeft(Node NewNode){
       LeftNode = NewNode;
}
public void SetRight(Node NewNode){
       RightNode = NewNode;
}
```

Question	Answer	Marks
3(b)	1 mark each • Creating 1 instance of `Node` with a correct value and storing the node … • … remaining 4 correct	**2**

Example program code:

Python

```
FirstNode = Node(10)
SecondNode = Node(20)
ThirdNode = Node(5)
FourthNode = Node(15)
FifthNode = Node(7)
```

VB.NET

```
Dim FirstNode As Node = New Node(10)
Dim SecondNode As Node = New Node(20)
Dim ThirdNode As Node = New Node(5)
Dim FourthNode As Node = New Node(15)
Dim FifthNode As Node = New Node(7)
```

Java

```
Node FirstNode = new Node(10);
Node SecondNode = new Node(20);
Node ThirdNode = new Node(5);
Node FourthNode = new Node(15);
Node FifthNode = new Node(7);
```

Question	Answer	Marks
3(c)(i)	1 mark each • Class `Tree` header (and end) no inheritance and constructor header (and end) taking 1 node parameter within class … • … storing parameter in `FirstNode` declared as a `Node` data type	**2**

Example program code:

Python

```
class Tree:
    def _init_(self, FirstNode):
        self._FirstNode = FirstNode #node
```

VB.NET

```
Class Tree
   Private FirstNode As Node
   Sub New(pFirstNode)
      FirstNode = pFirstNode
   End Sub
End Class
```

Java

```
class Tree{
       private Node FirstNode;

       public Tree(Node pFirstNode){
          FirstNode = pFirstNode;
       }
}
```

Question	Answer	Marks
3(c)(ii)	1 mark for • Get method header (and end) with no parameter, returning `FirstNode`	1

Example program code:

Python
```
def GetRootNode(self):
    return self._FirstNode
```

VB.NET
```
Function GetRootNode()
   Return FirstNode
End Function
```

Java
```
public Node GetRootNode(){
       return FirstNode;
}
```

Question	Answer	Marks
3(c)(iii)	1 mark each to max 6 • Insert method header (and end) taking 1 node parameter • If parameter < first node, checking if there is a left node … • … storing node in left node if it is null • If parameter >= first node, checking if there is a right node … • … storing node in right node if it is null • Looping until correct position is found // recursive calls	**6**

Question	Answer	Marks

Example program code:

Python

```
def Insert(self, NewNode):
    CurrentNode = self._FirstNode
    Inserted = True
    while Inserted:
        if NewNode.GetData() < CurrentNode.GetData():
            if CurrentNode.GetLeft() == None:
                CurrentNode.SetLeft(NewNode)
                return True
            else:
                CurrentNode = CurrentNode.GetLeft()
        else:
            if CurrentNode.GetRight() == None:
                CurrentNode.SetRight(NewNode)
                return True
            else:
                CurrentNode = CurrentNode.GetRight()
```

VB.NET

```
Function Insert(NewNode)
   Dim CurrentNode As Node
   CurrentNode = FirstNode
   Dim Inserted As Boolean = True
   While Inserted
      If NewNode.GetData() < CurrentNode.GetData() Then
         If CurrentNode.GetLeft() Is Nothing Then
            CurrentNode.SetLeft(NewNode)
            Return True
         Else
            CurrentNode = CurrentNode.GetLeft()
         End If
```

Question	Answer	Marks

```
      Else
         If CurrentNode.GetRight() Is Nothing Then
            CurrentNode.SetRight(NewNode)
            Return True
         Else
            CurrentNode = CurrentNode.GetRight()
         End If
      End If
   End While
End Function
```

Java

```
public Boolean Insert(Node NewNode){
       Node CurrentNode = FirstNode;
       Boolean Inserted = true;
       while(Inserted){
           if(NewNode.GetData() < CurrentNode.GetData()){
                if(CurrentNode.GetLeft() == null){
                      CurrentNode.SetLeft(NewNode);
                      return true;
                }else{
                      CurrentNode = CurrentNode.GetLeft();
                }
        }else{
                if(CurrentNode.GetRight() == null){
                      CurrentNode.SetRight(NewNode);
                      return true;
                }else{
                      CurrentNode = CurrentNode.GetRight();
                }
        }
    }
    return false;
}
```

Question	Answer	Marks
3(d)	1 mark each • Procedure header (and end) taking node as parameter, that is **recursive** • Checking if left is null and **recursive** call if not null • Outputting node's data • Checking if right is null and **recursive** call if not null • Correct order	**5**

Question	Answer	Marks

Example program code:

Python

```
def OutputInOrder(RootNode):
    if RootNode.GetLeft() != None:
        OutputInOrder(RootNode.GetLeft())
    print(RootNode.GetData())
    if RootNode.GetRight() != None:
        OutputInOrder(RootNode.GetRight())
```

VB.NET

```
Sub OutputInOrder(RootNode)
   If RootNode.GetLeft() IsNot Nothing Then
      OutputInOrder(RootNode.GetLeft())
   End If
   Console.WriteLine(RootNode.GetData())
   If RootNode.GetRight() IsNot Nothing Then
      OutputInOrder(RootNode.GetRight())
   End If
End Sub
```

Java

```
public static void OutputInOrder(Node RootNode){
       if(RootNode.GetLeft() != null){
           OutputInOrder(RootNode.GetLeft());
       }
       System.out.println(RootNode.GetData());
       if(RootNode.GetRight() != null){
           OutputInOrder(RootNode.GetRight());
       }
}
```

3(e)(i)	1 mark each • Creation of `Tree` object with the Node with value 10 as parameter • Calling method `Insert()` for tree with the nodes for 20, 5, 15 and 7 in order • Calling `OutputInOrder()` with tree's root node as parameter	3

Question	Answer	Marks

Example program code:

Python

```
FirstNode = Node(10)
SecondNode = Node(20)
ThirdNode = Node(5)
FourthNode = Node(15)
FifthNode = Node(7)
MyTree = Tree(FirstNode)
MyTree.Insert(SecondNode)
MyTree.Insert(ThirdNode)
MyTree.Insert(FourthNode)
MyTree.Insert(FifthNode)
OutputInOrder(MyTree.GetRootNode())
```

VB.NET

```
Sub Main(args As String())
    Dim FirstNode As Node = New Node(10)
    Dim SecondNode As Node = New Node(20)
    Dim ThirdNode As Node = New Node(5)
    Dim FourthNode As Node = New Node(15)
    Dim FifthNode As Node = New Node(7)
    Dim MyTree As Tree = New Tree(FirstNode)
    MyTree.Insert(SecondNode)
    MyTree.Insert(ThirdNode)
    MyTree.Insert(FourthNode)
    MyTree.Insert(FifthNode)
    OutputInOrder(MyTree.GetRootNode())
End Sub
```

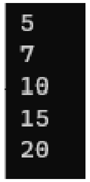
Question	Answer	Marks

Java

```
public static void main(String args[]){
       Node FirstNode = new Node(10);
       Node SecondNode = new Node(20);
       Node ThirdNode = new Node(5);
       Node FourthNode = new Node(15);
       Node FifthNode = new Node(7);

       Tree MyTree = new Tree(FirstNode);
       MyTree.Insert(SecondNode);
       MyTree.Insert(ThirdNode);
       MyTree.Insert(FourthNode);
       MyTree.Insert(FifthNode);
       OutputInOrder(MyTree.GetRootNode());
}
```

Question	Answer	Marks
3(e)(ii)	Output of 5 7 10 15 20	1



```
5
7
10
15
20
```