CAMBRIDGE
International Education

Cambridge International AS & A Level

COMPUTER SCIENCE **9618/43**

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 **45** 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) `Queue` array with 50 integer elements … • … all initialised to `−1` • (Global) `HeadPointer` and `TailPointer` initialised with `−1`	**3**

Question	Answer	Marks

Example program code:

Python

```
Queue = [] #integer 50 elements
HeadPointer = -1
TailPointer = -1
#main
HeadPointer = -1
TailPointer = -1
for x in range(50):
    Queue.append(-1)
```

VB.NET

```
Dim Queue(49) As Integer
Dim HeadPointer As Integer
Dim TailPointer As Integer
Sub Main(args As String())
   HeadPointer = -1
   TailPointer = -1
   For x = 0 To 49
      Queue(x) = -1
   Next
End Sub
```

Java

```
public static Integer[] Queue = new Integer[50];
public static Integer HeadPointer;
public static Integer TailPointer;
public static void main(String args[]){
    HeadPointer = -1;
    TailPointer = -1;
    for(Integer x = 0; x < 50; x++){
         Queue[x] = -1;
    }
}
```

Question	Answer	Marks
1(b)	1 mark each • Function `Enqueue()` header (and end) taking one (integer) parameter • Checking if `Queue` is full … • …returning `FALSE` if full and `TRUE` if not full • (Otherwise) storing data item at `TailPointer + 1` (check incrementing) • Incrementing `TailPointer` • Checking if this is the first element and incrementing/storing 0 in `HeadPointer`	**6**

Question	Answer	Marks

Example program code:

Python
```
def Enqueue(Data):
    global Queue
    global TailPointer
    global HeadPointer

    if TailPointer < 49:
        TailPointer = TailPointer + 1
        Queue[TailPointer] = Data

        if HeadPointer == -1:
            HeadPointer = 0
        return True
    else:
        return False
```

VB.NET
```
Function Enqueue(DataValue As Integer)
   If TailPointer < 49 Then
      TailPointer = TailPointer + 1
      Queue(TailPointer) = DataValue
      If HeadPointer = -1 Then
         HeadPointer = 0
      End If
      Return True
   Else
      Return False
   End If
End Function
```

Question	Answer	Marks

Java
```
public static Boolean Enqueue(Integer DataValue){
   if (TailPointer < 49){
        TailPointer++;
        Queue[TailPointer] = DataValue;
        if(HeadPointer == -1){
            HeadPointer = 0;
        }
        return true;
    }else{
        return false;
    }
}
```

Question	Answer	Marks
1(c)	1 mark each • Function `Dequeue()` header (and close) and returning appropriate value in all cases. • Checking if queue is empty … • … and returning `-1` if empty • Accessing and returning element at `Queue[HeadPointer]` (before `HeadPointer` is incremented) • Incrementing `HeadPointer`	**5**

Question	Answer	Marks

Example program code:

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

VB.NET
```
Function Dequeue()
   Dim ReturnValue As Integer
   If HeadPointer > -1 And HeadPointer <= TailPointer Then
      ReturnValue = Queue(HeadPointer)
      HeadPointer = HeadPointer + 1
      Return ReturnValue
   Else
      Return -1
   End If
End Function
```

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

Question	Answer	Marks
1(d)	1 mark each • `CreateQueue()` header (and end) **and** opening the file to read **and** closing file in appropriate place • Looping until end of file • Reading in each/all lines (and converting to integer and removing new line) • … calling `Enqueue()` **once** with **each** value … • … checking return value and outputting `"Queue full"` if full (can output once or many times) • Exception try, catch with appropriate output. All file access within try	**6**

Question	Answer	Marks

Example program code:

Python

```
def CreateQueue():
    try:
        File = open("QueueData.txt")
        for Line in File:
            ReturnValue = Enqueue(int(Line))
            if ReturnValue == False:
                print("Queue full")
                break;
        File.close()
    except:
        print("Cannot open or read file")
```

VB.NET

```
Sub CreateQueue()
   Dim ReturnValue As Boolean
   Dim ReadData As Integer
   Try
      Dim FileReader As New System.IO.StreamReader("QueueData.txt")
      While Not FileReader.EndOfStream
         ReadData = FileReader.ReadLine()
         ReturnValue = Enqueue(ReadData)
         If ReturnValue = False Then
            Console.WriteLine("Queue full")
         End If
      End While
      FileReader.Close()
   Catch ex As Exception
      Console.WriteLine("Cannot open or read file")
   End Try
End Sub
```

Question	Answer	Marks

Java

```
public static void CreateQueue(){
     Boolean ReturnValue;
     Integer ReadData;
     try{
           FileReader f = new FileReader("QueueData.txt");
           try{
                BufferedReader Reader = new BufferedReader(f);
                String Line = Reader.readLine();
                Line = Line.replace("\n","");
                while (Line != null){
                    Line = Line.replace("\n","");
                    ReturnValue = Enqueue(Integer.parseInt(Line));
                    if (ReturnValue == false){
                         System.out.println("Queue full");
                    }
                    Line = Reader.readLine();
                }
                Reader.close();
           }catch(IOException ex){
           }
    }catch(FileNotFoundException e){ System.out.println("Cannot open or read file");}
}
```

Question	Answer	Marks
1(e)(i)	1 mark each • Calling `CreateQueue()` • Calling `Dequeue()` and storing/using return value … • … repeatedly until return value is `-1` • … adding together all **return values** to create a total within the loop … • … outputting the total	**5**

Example program code:

Python

```
CreateQueue()
Total = 0
ReturnValue = 0


while ReturnValue > -1:
    ReturnValue = Dequeue()
    if ReturnValue != -1:
        Total = Total + ReturnValue
print("The total is", Total)
```

Question	Answer	Marks

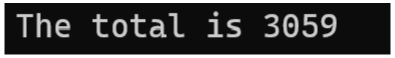
VB.NET

```
CreateQueue()
Dim Total As Integer = 0
Dim ReturnValue As Integer = 0
While ReturnValue > -1
   ReturnValue = Dequeue()
   If ReturnValue <> -1 Then
      Total = Total + ReturnValue
   End If
End While
Console.WriteLine("The total is " & Total)
```

Java

```
CreateQueue();
Integer Total = 0;
Integer ReturnValue = 0;
while(ReturnValue > -1){
     ReturnValue = Dequeue();
     if(ReturnValue != -1){
          Total = Total + ReturnValue;
     }
}
System.out.println("The total is " + Total);
```

1(e)(ii)	1 mark for screenshot showing 3059	1



```
The total is 3059
```

Question	Answer	Marks
2(a)	1 mark for array declared with data values: 0 3 4 56 67 44 43 32 31 345 45 6 54 1	**1**

Example program code:

Python
```
DataArray = [0, 3, 4, 56, 67, 44, 43, 32, 31, 345, 45, 6, 54, 1]
```

Java
```
Integer[] DataArray = {0,3,4,56,67,44,43,32,31,345,45,6,54,1};
```

VB.NET
```
Dim DataArray() As Integer = {0, 3, 4, 56, 67, 44, 43, 32, 31, 345, 45, 6, 54, 1}
```

Question	Answer	Marks
2(b)	1 mark each • `InsertionSort()` header (and close) taking array as a parameter and returning (attempt at) sorted array • Looping through/for each element • Extracting element and comparing to sorted list … • … moving elements in sorted list • … and inserting element in correct position (ascending order)	**5**

Question	Answer	Marks

Example program code:

Python

```
def InsertionSort(DataArray):
    if (len(DataArray)) <= 1:
        return DataArray
    for X in range(1, len(DataArray)):
        CurrentValue = DataArray[X]
        Y = X-1
        while Y >=0 and CurrentValue < DataArray[Y]:
            DataArray[Y+1] = DataArray[Y]
            Y = Y -1
        DataArray[Y+1] = CurrentValue
    return DataArray
```

Java

```
public static Integer[] InsertionSort(Integer[] DataArray){
     Integer CurrentValue = 0;
     Integer Y = 0;
     if(DataArray.length <= 1){
          return DataArray;
     }
     for(Integer X = 1; X <= DataArray.length -1; X++){
          CurrentValue = DataArray[X];
          Y = X -1;
          while(Y >= 0 && CurrentValue < DataArray[Y]){
                DataArray[Y + 1] = DataArray[Y];
                Y--;
          }
          DataArray[Y+1] = CurrentValue;
     }
     return DataArray;
}
```

Question	Answer	Marks

VB.NET

```
Function InsertionSort(DataArray)
   Dim CurrentValue As Integer
   Dim Y As Integer
   If (DataArray.length()) <= 1 Then
      Return DataArray
   End If
   For X = 1 To DataArray.length() - 1
      CurrentValue = DataArray(X)
      Y = X - 1
      While Y >= 0 AndAlso CurrentValue < DataArray(Y)
         DataArray(Y + 1) = DataArray(Y)
         Y = Y - 1
      End While
      DataArray(Y + 1) = CurrentValue
   Next X
   Return DataArray
End Function
```

Question	Answer	Marks
2(c)	1 mark each • `OutputArray()` header (and close) taking an array parameter and outputting the array contents … • … in correct format	**2**

Question	Answer	Marks

Example program code:

Python
```
def OutputArray(DataArray):
    Output = ""
    for Item in DataArray:
        Output = Output + str(Item) + " "
    print(Output)
```

Java
```
public static void OutputArray(Integer[] DataArray){
     String Output = "";
     Integer X = 0;
     while(X < DataArray.length){
          if(DataArray[X] != -1){
               Output = Output + DataArray[X] + " ";
          }
          X = X + 1;
     }
     System.out.println(Output);
}
```

VB.NET
```
Sub OutputArray(DataArray)
   Dim Output As String = ""
   Dim X As Integer = 0
   While X < DataArray.length
      If DataArray(X) <> -1 Then
         Output = Output & DataArray(X) & " "
      End If
      X = X + 1
   End While
   Console.WriteLine(Output)
End Sub
```

Question	Answer	Marks
2(d)(i)	1 mark each • Calling `InsertionSort()` with array parameter **and** storing/using return array • … calling `OutputArray()` with array parameter before and after `InsertionSort()`	**2**

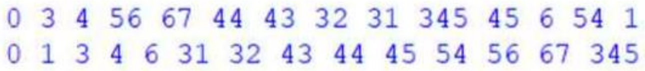
Example program code

Python
```
OutputArray(DataArray)
DataArray = InsertionSort(DataArray)
OutputArray(DataArray)
```

Java
```
OutputArray(DataArray);
DataArray = InsertionSort(DataArray);
OutputArray(DataArray);
```

VB.NET
```
OutputArray(DataArray)
DataArray = InsertionSort(DataArray)
OutputArray(DataArray)
```

Question	Answer	Marks

2(d)(ii)	1 mark for output showing unsorted then sorted array
	
	e.g.
	
	`0 3 4 56 67 44 43 32 31 345 45 6 54 1`
	`0 1 3 4 6 31 32 43 44 45 54 56 67 345`
		1

Question	Answer	Marks
2(e)	1 mark each • `Search()` header (and close) taking array and integer as parameters • Looping/recursive calls until no elements left/Low<=High **and** returning –1 if not found • … calculating middle index and accessing this value • … comparison of array at middle value to integer parameter • … if they are equal return mid • … if array[mid] < parameter update low to middle + 1, if array[mid] > parameter update high to middle – 1 // recursive call with updated low and updated high	**6**

Question	Answer	Marks

Example program code:

Python

```
def Search(DataArray, ItemToFind):
    Low = 0
    High = len(DataArray) - 1
    Middle = 0

    while Low <= High:
        Middle = (High + Low) // 2
        if DataArray[Middle] < ItemToFind:
            Low = Middle + 1
        elif DataArray[Middle] > ItemToFind:
            High = Middle - 1
        else:
            return Middle
    return -1
```

Java

```
public static Integer Search(Integer[] DataArray, Integer ItemToFind){
     Integer Low = 0;
     Integer High = DataArray.length - 1;
     Integer Middle = 0;
     while(Low <= High){
          Middle = (High + Low) / 2;
          if(DataArray[Middle] < ItemToFind){
               Low = Middle + 1;
          }else if(DataArray[Middle] > ItemToFind){
               High = Middle - 1;
          }else{
               return Middle;
          }
     }
     return -1;
}
```

Question	Answer	Marks

VB.NET

```
Function Search(DataArray, ItemToFind)
   Dim Low As Integer = 0
   Dim High As Integer = DataArray.length() - 1
   Dim Middle As Integer = 0
   While Low <= High
      Middle = (High + Low) \ 2
      If DataArray(Middle) < ItemToFind Then
         Low = Middle + 1
      ElseIf DataArray(Middle) > ItemToFind Then
         High = Middle - 1
      Else
         Return Middle
      End If
   End While
   Return -1
End Function
```

Question	Answer	Marks
2(f)(i)	1 mark each • Calling `Search()` with all four sets of values 0 345 67 2 • … outputting 'not found' or index in appropriate message each time	**2**

Question	Answer	Marks

Example program code:

Python

```
Location = Search(DataArray, 0)
if Location == -1:
    print("Data not found")
else:
    print("Data found at", Location)
Location = Search(DataArray, 345)
if Location == -1:
    print("Data not found")
else:
    print("Data found at", Location)

Location = Search(DataArray, 67)
if Location == -1:
    print("Data not found")
else:
    print("Data found at", Location)
Location = Search(DataArray, 2)
if Location == -1:
    print("Data not found")
else:
    print("Data found at", Location)
```

Java

```
Integer Location = Search(DataArray,0);
if(Location == -1){
     System.out.println("Data not found");
}else{
     System.out.println("Data found at " + Location);
}
Location = Search(DataArray,345);
```

Question	Answer	Marks

```
if(Location == -1){
     System.out.println("Data not found");
}else{
     System.out.println("Data found at " + Location);
}
Location = Search(DataArray,67);
if(Location == -1){
     System.out.println("Data not found");
}else{
     System.out.println("Data found at " + Location);
}
Location = Search(DataArray,2);
if(Location == -1){
     System.out.println("Data not found");
}else{
     System.out.println("Data found at " + Location);
}
```

VB.NET

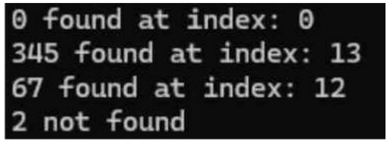
```
Dim Location As Integer = Search(DataArray, 0)
If Location = -1 Then
   Console.WriteLine("Data not found")
Else
   Console.WriteLine("Data found at " & Location)
End If
```

Question	Answer	Marks

```
Location = Search(DataArray, 345)
If Location = -1 Then
   Console.WriteLine("Data not found")
Else
   Console.WriteLine("Data found at " & Location)
End If
Location = Search(DataArray, 67)
If Location = -1 Then
   Console.WriteLine("Data not found")
Else
   Console.WriteLine("Data found at " & Location)
End If
Location = Search(DataArray, 2)
If Location = -1 Then
   Console.WriteLine("Data not found")
Else
   Console.WriteLine("Data found at " & Location)
End If
```

Question	Answer	Marks
2(f)(ii)	1 mark for output showing locations for first 3 and not found for 4th	1

e.g.



```
0 found at index: 0
345 found at index: 13
67 found at index: 12
2 not found
```

Question	Answer	Marks
3(a)(i)	1 mark each • Class `Node` header (and end) • `TheData` declared as Integer, `NextNode` declared as `Node` • Constructor header (and end) taking (min) 1 parameter … • … storing parameter to `TheData` within constructor and storing null value to `NextNode` within constructor	**4**

Example program code:

Python
```
class Node:
    def _init_(self, NodeData):
        self._TheData = NodeData #Integer
        self._NextNode = None #Node
```
Java
```
class Node{
     public Integer TheData;
     public Node NextNode;

     public Node(Integer NodeData){
          TheData = NodeData;
          NextNode = null;
    }}
```

VB.NET
```
Class Node
   Public TheData As Integer
   Public NextNode As Node
   Sub New(NodeData)
      TheData = NodeData
      NextNode = Nothing
   End Sub
End Class
```

Question	Answer	Marks
3(a)(ii)	1 mark each • 1 get method header (and close) taking no parameters … • … returning correct value (without overwriting) • 2nd correct get method	**3**

Example program code:

Python

```
def GetData(self):
    return self._TheData
def GetNextNode(self):
    return self._NextNode
```

Java

```
public Integer GetData(){
       return TheData;
}
public Node GetNextNode(){
       return NextNode;
}
```

VB.NET

```
Function GetData()
   Return TheData
End Function
Function GetNextNode()
   Return NextNode
End Function
```

Question	Answer	Marks
3(a)(iii)	1 mark each • `SetNextNode()` method header (and close) taking 1 parameter (of type `Node`) … • … storing parameter in `NextNode`	**2**

Example program code:

Python

```
def SetNextNode(self, pNextNode):
    self._NextNode = pNextNode
```

Java

```
public void SetNextNode(Node pNextNode){
     NextNode = pNextNode;
}
```

VB.NET

```
Sub SetNextNode(pNextNode)
   NextNode = pNextNode
End Sub
```

Question	Answer	Marks
3(b)(i)	1 mark each - Class `LinkedList` header (and close) and constructor header with no parameter (and close) … - … declaring `HeadNode` as type `Node` and storing null value in constructor	**2**

Example program code:

Python

```
class LinkedList:
    def _init_(self):
        self._HeadNode = None #Node
```

Java

```
class LinkedList{
      public Node HeadNode;
      public LinkedList(){
          HeadNode = null;
      }}
```

VB.NET

```
Class LinkedList
   Private HeadNode As Node
   Sub New()
      HeadNode = Nothing
   End Sub
End Class
```

Question	Answer	Marks
3(b)(ii)	1 mark each • `InsertNode()` method header (and close) taking one (integer) parameter • Creating new instance of `Node` with the parameter as the argument • Calling `SetNextNode()` for new node with `HeadNode` as parameter • Replacing `HeadNode` with new node	4

Example program code:

Python

```
def InsertNode(self, NodeData):
    TheNode = Node(NodeData)
    TheNode.SetNextNode(self._HeadNode)
    self._HeadNode = TheNode
```

Java

```
public void InsertNode(Integer NodeData){
     Node TheNode = new Node(NodeData);
     TheNode.SetNextNode(HeadNode);
     HeadNode = TheNode;
}
```

VB.NET

```
Sub InsertNode(NodeData)
   Dim TheNode As Node = New Node(NodeData)
   TheNode.SetNextNode(HeadNode)
   HeadNode = TheNode
End Sub
```

Question	Answer	Marks
3(b)(iii)	1 mark each • `Traverse()` method header (and close) with no parameter and returns created string • Starts at head node and follows nodes using `GetNextNode()` until no nodes left … • … concatenates the data from each node and formats correctly	**3**

Question	Answer	Marks

Example program code:

Python

```
def Traverse(self):
    ReturnValue = ""
    CurrentNode = self._HeadNode
    while(CurrentNode != None):
        ReturnValue = ReturnValue + str(CurrentNode.GetData())+ " "
        CurrentNode = CurrentNode.GetNextNode()
    return ReturnValue
```

Java

```
public String Traverse(){
     String ReturnValue = "";
     Node CurrentNode = new Node(-1);
     CurrentNode = HeadNode;
     while(CurrentNode != null){
          ReturnValue = ReturnValue + CurrentNode.GetData() + " ";
          CurrentNode = CurrentNode.GetNextNode();
     }
     return ReturnValue;
}
```

VB.NET

```
Function Traverse()
   Dim ReturnValue As String = ""
   Dim CurrentNode As Node = HeadNode

   While CurrentNode IsNot Nothing
      ReturnValue = ReturnValue & CurrentNode.GetData() & " "
      CurrentNode = CurrentNode.GetNextNode()
   End While
   Return ReturnValue
End Function
```

Question	Answer	Marks
3(b)(iv)	1 mark each to max 6 • `RemoveNode()` method header (and close) taking (integer) parameter and returning Boolean in all cases • Checking if head node is null and returning `FALSE` • Checking if head node equals parameter and returning `TRUE` if true … • … and updating `HeadNode` to `HeadNode.GetNextNode()` • Following nodes comparing data from each node to parameter … • ... if found updating next node and returning `TRUE` • … if end of list returning `FALSE`	**6**

Question	Answer	Marks

Example program code:

Python

```
def RemoveNode(self, DataToRemove):
    if self._HeadNode == None:
        return False
    elif self._HeadNode.GetData() == DataToRemove:
        self._HeadNode = self._HeadNode.GetNextNode()
        return True
    Found = False
    CurrentNode = self._HeadNode
    while not(Found) and CurrentNode != None:
        if ((CurrentNode).GetNextNode()).GetData() == DataToRemove:
            CurrentNode.SetNextNode(CurrentNode.GetNextNode().GetNextNode())
            Found = True
        else:
            CurrentNode = CurrentNode.GetNextNode()
```

Java

```
public Boolean RemoveNode(Integer DataToRemove){
     if(HeadNode == null){
          return false;
     }else if(HeadNode.GetData().equals(DataToRemove)){
          HeadNode = HeadNode.GetNextNode();
          return true;
     }
     Boolean Found = false;
     Node CurrentNode = new Node(-1);
     CurrentNode = HeadNode;
     Node NextNode = new Node(-1);
     while(! Found && CurrentNode != null){
          NextNode = CurrentNode.GetNextNode();
          if(NextNode.GetData().equals(DataToRemove)){
               CurrentNode.SetNextNode(NextNode.GetNextNode());
               return true;
```

Question	Answer	Marks

```
          }else{
               CurrentNode = CurrentNode.GetNextNode();
          }
     }
     return false;
}
```

VB.NET

```
Function RemoveNode(DataToRemove)
   If HeadNode Is Nothing Then

      Return False
   ElseIf HeadNode.GetData() = DataToRemove Then
      HeadNode = HeadNode.GetNextNode()
      Return True
   End If
   Dim Found As Boolean = False
   Dim CurrentNode As Node = HeadNode
   While Not (Found) And CurrentNode IsNot Nothing


      If ((CurrentNode).GetNextNode()).GetData() = DataToRemove Then
         CurrentNode.SetNextNode(CurrentNode.GetNextNode().GetNextNode())
         Found = True
      Else
         CurrentNode = CurrentNode.GetNextNode()
      End If
   End While
   Return Found
End Function
```

Question	Answer	Marks
3(c)(i)	1 mark each • Creating new `LinkedList` object • Calling `InsertNode()` five times with correct data in correct order • Calling `RemoveNode(30)` and calling `Traverse()` and store/output the return value, before `RemoveNode()` and after Full marks can be awarded to students who may have stored and/or outputted the return value from the function call.	**3**

Question	Answer	Marks

Example program code:

Python
```
CreateList = LinkedList()

CreateList.InsertNode(10)
CreateList.InsertNode(20)
CreateList.InsertNode(30)
CreateList.InsertNode(40)
CreateList.InsertNode(50)
ReturnValue1 = (CreateList.Traverse())
CreateList.RemoveNode(30)
ReturnValue2 = (CreateList.Traverse())
```

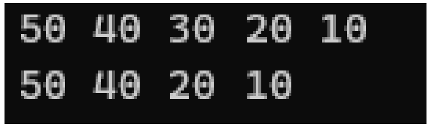
Java
```
public static void main(String args[]){
     LinkedList CreateList = new LinkedList();
     String ReturnValue2;
     String ReturnValue1;

     CreateList.InsertNode(10);
     CreateList.InsertNode(20);
     CreateList.InsertNode(30);
     CreateList.InsertNode(40);
     CreateList.InsertNode(50);
     ReturnValue1 = (CreateList.Traverse());
     CreateList.RemoveNode(30);
     ReturnValue2 = (CreateList.Traverse());
}
```

VB.NET
```
Sub Main(args As String())
   Dim CreateList As LinkedList = New LinkedList()
   Dim ReturnValue1 As String
   Dim ReturnValue2 As String
```

```
    CreateList.InsertNode(10)
    CreateList.InsertNode(20)
    CreateList.InsertNode(30)
    CreateList.InsertNode(40)
    CreateList.InsertNode(50)
    ReturnValue1 = (CreateList.Traverse())
    CreateList.RemoveNode(30)
    ReturnValue2 = (CreateList.Traverse())
    Console.ReadLine()
End Sub
```

Question	Answer	Marks
3(c)(ii)	1 mark each • Output of linked list with 50 40 30 20 10 • Output of 2nd linked list with 50 40 20 10	**2**

e.g.



```
50 40 30 20 10
50 40 20 10
```