

A dark blue vertical bar on the left side of the page. A blue arrow points to the right from the bar, containing the date.

9/7/2024

DataStructures

Lab 3 Exercise Solutions

Several thin, curved lines in dark blue and light grey originate from the bottom left and curve upwards and to the right.

Uzair Ali S/O Yar Muhammad Memon
B-S CS III Section A
023-23-0350
To: Ma'am Marina Rajpoot

Task #1

Solution:

The image shows two screenshots of the Visual Studio Code editor. The top screenshot displays the `Node.java` file, which defines a `Node` class with attributes `String name`, `Node next`, and `Node prev`. It includes a constructor `Node(String name)` that initializes these fields. The bottom screenshot shows the `DoubleLinkedList.java` file, which implements a `DoubleLinkedList` class. It features a static `head` variable and two methods: `insertAtBeginning(String name)` and `insertAtBeginning(String name, Node head)`. The first method handles the initial insertion when the list is empty, while the second method inserts a new node at the beginning of an existing list by updating the `next` and `prev` pointers of the existing nodes.

```
1 public class Node{
2     String name;
3     Node next;
4     Node prev;
5     public Node(String name){
6         this.name=name;
7         this.next=null;
8         this.prev=null;
9     }
10 }
```

```
1 public class DoubleLinkedList {
2     public static Node head;
3     public static void insertAtBeginning(String name)
4     {
5         Node newNode=new Node(name);
6         if(head==null){
7             head=newNode;
8         }
9         else{
10
11             newNode.next=head;
12             head.prev=newNode;
13             head=newNode;
14         }
15     }
16     public static Node insertAtBeginning(String name,Node head)
17     {
18         Node newNode=new Node(name);
19         if(head==null){
20             head=newNode;
21             return head;
22         }
23         else{
24             Node temp=head;
25             newNode.next=temp;
26             temp.prev=newNode;
27             head=newNode;
28             return head;
29         }
30     }
31 }
```

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project named 'LAB3'. The file 'DoubleLinkedList.java' is selected and open in the editor. The code in the editor is as follows:

```
31 public static void insertAtEnd(String name)
32 {
33     Node newNode=new Node(name);
34     if(head==null){
35         head=newNode;
36     }
37     else{
38         Node temp=head;
39         while(temp.next!=null){
40             temp=temp.next;
41         }
42         temp.next=newNode;
43         newNode.prev=temp;
44     }
45 }
46
47 public static Node insertAtEnd(String name,Node head1)
48 {
49     Node newNode=new Node(name);
50     if(head1==null){
51         head1=newNode;
52         return head1;
53     }
54     else{
55         Node temp=head1;
56         while(temp.next!=null){
57             temp=temp.next;
58         }
59         temp.next=newNode;
60         newNode.prev=temp;
61         return head1;
62 }
```

The status bar at the bottom indicates 'Ln 17, Col 3', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system tray shows a temperature of 36°C and the time 2:30 PM on 9/7/2024.

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project named 'LAB3'. The file 'DoubleLinkedList.java' is selected and open in the editor. The code in the editor is as follows:

```
61 }
62 }
63
64 /*public static void Print(Node head){
65     if (head==null){
66         System.out.println("Linked List is empty");
67     }
68     else{
69         Node temp=head;
70         while(temp!=null){
71             System.out.print(temp.name+" ");
72             temp=temp.next;
73         }
74     }
75 }
76
77 public static void Print(){
78     if (head==null){
79         System.out.println("Linked List is empty");
80     }
81     else{
82         Node temp=head;
83         while(temp!=null){
84             System.out.print(temp.name+" ");
85             temp=temp.next;
86         }
87     }
88 }
89
90 public static Node insertAfterName(String name, Node head ,String name1)
91 {
92     Node newNode =new Node(name1);
93     int index=1;
```

The status bar at the bottom indicates 'Ln 17, Col 3', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system tray shows a temperature of 36°C and the time 2:30 PM on 9/7/2024.

This screenshot shows the implementation of the `insertAtBeginning` method in `DoubleLinkedList.java`. The Explorer pane on the left lists files under 'LAB3', including `CheckCycle.java`, `Double.java`, `DoubleLinkedList.java`, `DoubleWithTail.java`, `Lab 03 Sept-09-2024.pdf`, `Node.java`, `Node1.java`, and `SingleLinkedList.java`. The main editor displays the following code:

```
191 Node temp=head;
192 Node temp1=head;
193 while(temp1.next!=null){
194     if(temp1.name==name){
195         break;
196     }
197     index++;
198     temp1=temp1.next;
199 }
200 if(temp1.next==null&&temp1.name!=name){
201     System.out.println(x:"Name not found");
202     return head;
203 }
204
205 for(int i=1;i<index;i++)
206 {
207     temp=temp.next;
208 }
209 newNode.next=temp.next;
210 temp.next=newNode;
211 return head;
212 }
213 public static Node insertBeforeName(String name, Node head,String name1)
214 {
215     Node newNode =new Node(name1);
216     int index=1;
217     Node temp=head;
218     Node temp1=head;
219     while(temp1.next!=null){
220         if(temp1.name==name){
```

The status bar at the bottom indicates 'Ln 17, Col 3', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system clock shows 2:30 PM on 9/7/2024.

This screenshot shows the implementation of the `makeCircular` and `printAll` methods in `DoubleLinkedList.java`. The Explorer pane on the left is the same as in the previous screenshot. The main editor displays the following code:

```
221         break;
222     }
223     index++;
224     temp1=temp1.next;
225 }
226 if(temp1.next==null&&temp1.name!=name){
227     System.out.println(x:"Name not found");
228     return head;
229 }
230
231 for(int i=1;i<index-1;i++)
232 {
233     temp=temp.next;
234 }
235 newNode.next=temp.next;
236 temp.next=newNode;
237 return head;
238 }
239 // Make double linkedlist as Circular Double LinkedList
240 public static void makeCircular()
241 {
242     Node temp=head;
243     while(temp.next!=null){
244         temp=temp.next;
245     }
246     temp.next=head;
247     head.prev=temp;
248     Node temp1=head;
249 }
250 public static void printAll(Node head){
```

The status bar at the bottom indicates 'Ln 17, Col 3', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system clock shows 2:30 PM on 9/7/2024.

```
151 if(head==null){
152     System.out.println(x:"Linked list is empty");
153 }
154 else{
155     Node temp=head;
156     System.out.print(s:"Circular Linked List is : ");
157     while(temp.next!=head){
158         System.out.print(temp.name+" ");
159         temp=temp.next;
160     }
161     System.out.print(temp.name+" ");
162     System.out.println();
163 }
164 if(head1==null){
165     System.out.println(x:"Linked List is empty");
166 }
167 else{
168     System.out.print(s:"Other Linked list is : ");
169     Node temp1=head1;
170     while(temp1!=null){
171         System.out.print(temp1.name+" ");
172         temp1=temp1.next;
173     }
174 }
175 }
176 public static void main(String[] args) {
177     Node head1=null;
178     head1=insertAtBeginning(name:"uzair",head1);
179     head1=insertAtBeginning(name:"Zubair", head1);
```

```
180     head1=insertAtBeginning(name:"GM", head1);
181     insertAtBeginning(name:"Rehman");
182     insertAtBeginning(name:"Hussain");
183     insertAtEnd(name:"Memon");
184     head1=insertAtEnd(name:"Faizan", head1);
185     // Print();
186     System.out.println();
187     // Print(head1);
188     System.out.println();
189     head1=insertAfterName(name:"Faizan", head1, name:"Mustafa");
190     // Print(head1);
191     System.out.println();
192     head1=insertBeforeName(name:"Mustafa", head1, name:"Mujtaba");
193     //Print(head1);
194     makeCircular();
195     printAll(head1);
196 }
197 }
198 }
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

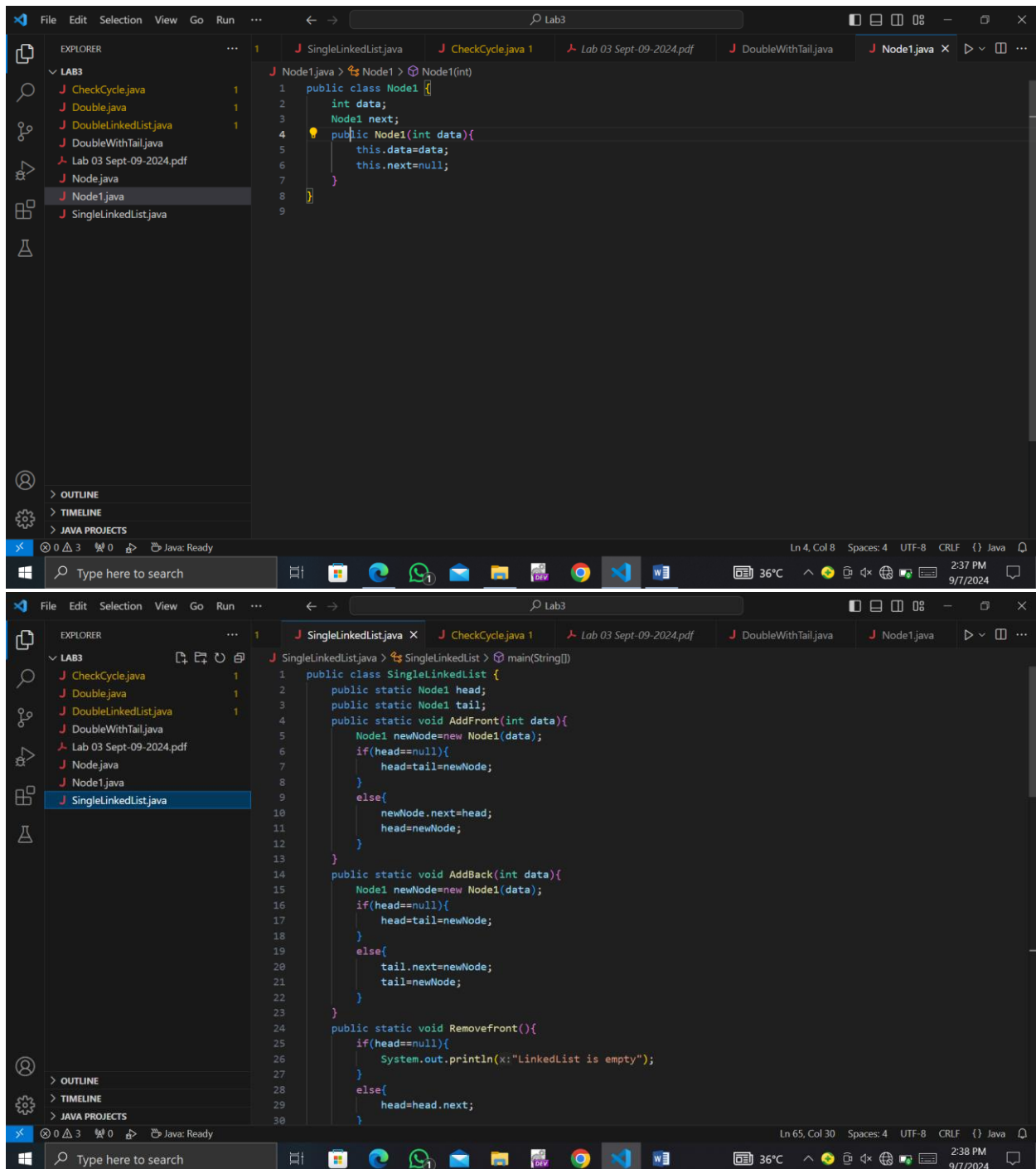
a\jdt_ws\Lab3_e944fd4d\bin' 'DoubleLinkedList'

powerShell
Run: Doubl...

Circular Linked List is : Hussain Rehman Memon
Other Linked list is : GM Zubair uzair Faizan Mujtaba Mustafa
PS E:\bscs3\DSA Lab\Lab3>

Task #2

Solution: Single Linked List with Tail



The image displays two screenshots of a Visual Studio Code editor window, showing the implementation of a Single Linked List with Tail in Java.

Top Screenshot: The editor shows the `Node1.java` file. The code defines a `Node1` class with an `int data` field and a `Node1 next` field. The `Node1(int data)` constructor initializes `this.data = data` and `this.next = null`.

```
1 public class Node1 {
2     int data;
3     Node1 next;
4     public Node1(int data){
5         this.data=data;
6         this.next=null;
7     }
8 }
9
```

Bottom Screenshot: The editor shows the `SingleLinkedList.java` file. The code defines a `SingleLinkedList` class with static fields `head` and `tail`, and static methods `AddFront`, `AddBack`, and `Removefront`. The `main` method is also present.

```
1 public class SingleLinkedList {
2     public static Node1 head;
3     public static Node1 tail;
4     public static void AddFront(int data){
5         Node1 newNode=new Node1(data);
6         if(head==null){
7             head=tail=newNode;
8         }
9         else{
10            newNode.next=head;
11            head=newNode;
12        }
13    }
14    public static void AddBack(int data){
15        Node1 newNode=new Node1(data);
16        if(head==null){
17            head=tail=newNode;
18        }
19        else{
20            tail.next=newNode;
21            tail=newNode;
22        }
23    }
24    public static void Removefront(){
25        if(head==null){
26            System.out.println("LinkedList is empty");
27        }
28        else{
29            head=head.next;
30        }
31    }
32 }
```

The screenshot shows an IDE with a project named 'LAB3'. The 'EXPLORER' pane on the left lists several files: 'CheckCycle.java', 'Double.java', 'DoubleLinkedList.java', 'DoubleWithTail.java', 'Lab 03 Sept-09-2024.pdf', 'Node.java', 'Node1.java', and 'SingleLinkedList.java'. The 'SingleLinkedList.java' file is open in the editor, showing the following code:

```
31 }
32 public static void Removelast(){
33     if(head==null){
34         System.out.println(x:"Linked list is empty");
35     }
36     else{
37         Node1 temp=head;
38         while(temp.next.next!=null){
39             temp=temp.next;
40         }
41         temp.next=null;
42         tail=temp;
43     }
44 }
45 public static void print(){
46     if(head==null){
47         System.out.println(x:"Linked List is empty");
48     }
49     else{
50         Node1 temp=head;
51         while(temp!=null){
52             System.out.print(temp.data+" ");
53             temp=temp.next;
54         }
55     }
56 }
57 Run | Debug
58 public static void main(String[] args) {
59     AddFront(data:5);
60     AddFront(data:4);
```

The status bar at the bottom indicates 'Ln 45, Col 27', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system tray shows the date and time as '2:38 PM 9/7/2024'.

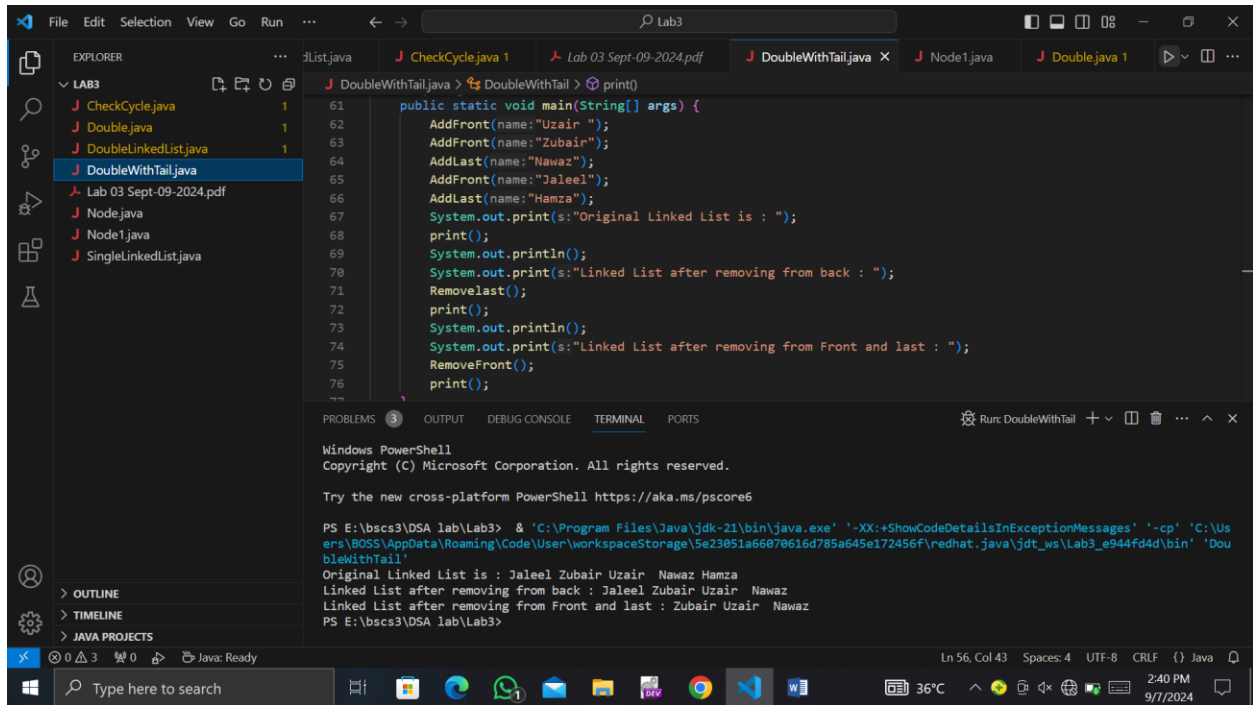
The screenshot shows the same IDE with the 'SingleLinkedList.java' file open. The code is now complete, including the 'main' method and the 'Removefront()' method. The 'TERMINAL' pane at the bottom shows the output of the program:

```
PS E:\bscs3\DSA lab\Lab3> & 'C:\Program Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\8055\AppData\Roaming\Code\User\workspaceStorage\5e23051a66070616d785a645e172456F\redhat.java\jdt_ws\Lab3_e944f44d\bin' 'SingleLinkedList'
Original Linked List is : 3 4 5 6 7
Linked List after removing from back : 3 4 5 6
Linked List after removing from Front and back : 4 5 6
PS E:\bscs3\DSA lab\Lab3>
```

The status bar at the bottom indicates 'Ln 45, Col 27', 'Spaces: 4', 'UTF-8', 'CRLF', and 'Java'. The system tray shows the date and time as '2:38 PM 9/7/2024'.

Double Linked List with Tail

```
1 public class DoubleWithTail {
2     public static Node head;
3     public static Node tail;
4     public static void AddFront(String name){
5         Node newNode=new Node(name);
6         if(head==null){
7             head=tail=newNode;
8         }
9         else{
10            newNode.next=head;
11            head.prev=newNode;
12            head=newNode;
13        }
14    }
15    public static void AddLast(String name){
16        Node newNode=new Node(name);
17        if(head==null){
18            head=tail=newNode;
19        }
20        else{
21            tail.next=newNode;
22            newNode.prev=tail;
23            tail=newNode;
24        }
25    }
26    public static void RemoveFront(){
27        if(head==null){
28            System.out.println(x:"Linked List is empty");
29        }
30        else{
31            head=head.next;
32            head.prev=null;
33        }
34    }
35    public static void RemoveLast(){
36        if(head==null){
37            System.out.println(x:"Linked List is empty");
38        }
39        else{
40            Node temp=head;
41            while(temp.next.next!=null){
42                temp=temp.next;
43            }
44            temp.next=null;
45            tail.prev=null;
46            tail=temp;
47        }
48    }
49    public static void print(){
50        if(head==null){
51            System.out.println(x:"Linked List is empty");
52        }
53        else{
54            Node temp=head;
55            while(temp!=null){
56                System.out.print(temp.name+" ");
57                temp=temp.next;
58            }
59        }
60    }
```

Task #3

Solution:

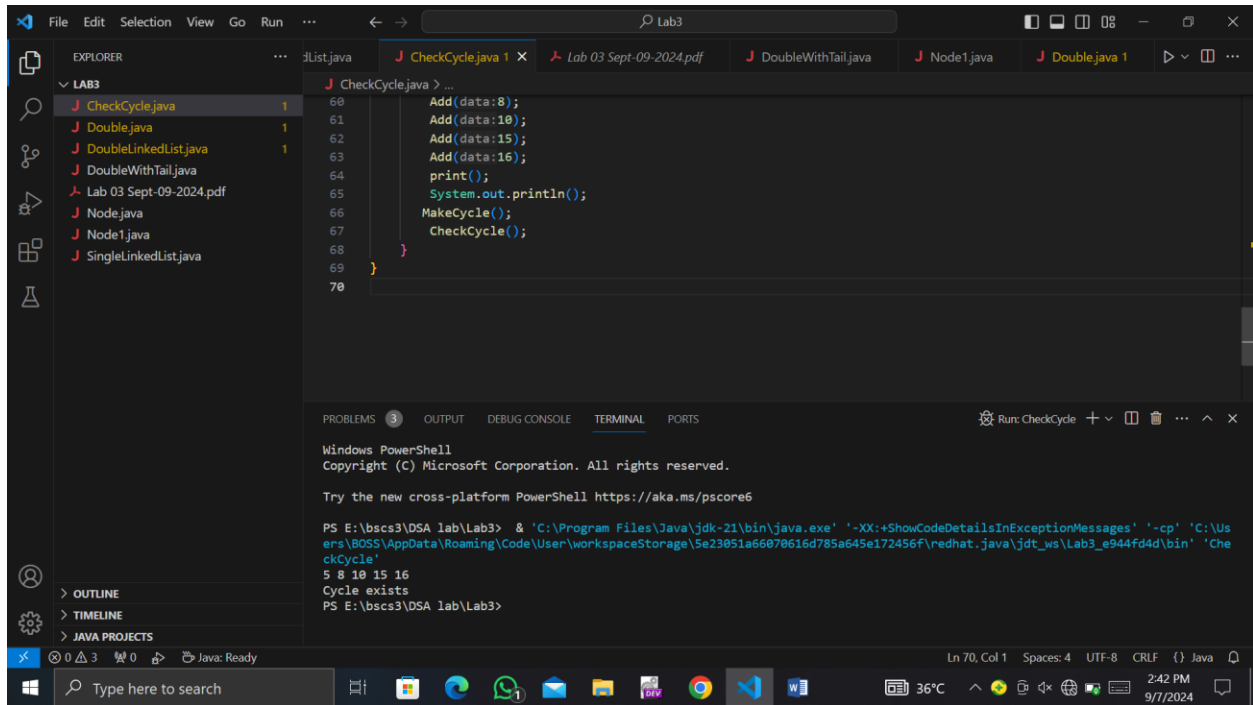
The image shows two screenshots of a code editor (likely IntelliJ IDEA) displaying the implementation of a linked list with cycle detection. The top screenshot shows the initial setup of the `CheckCycle` class, and the bottom screenshot shows the implementation of the `CheckCycle` method and the `main` method.

Top Screenshot:

```
1 public class CheckCycle {
2     public static Node1 head;
3     public static Node1 tail;
4     public static int size=0;
5     public static void Add(int data){
6         Node1 newNode=new Node1(data);
7         if(head==null){
8             head=tail=newNode;
9             size++;
10        }
11        else{
12            tail.next=newNode;
13            tail=newNode;
14            size++;
15        }
16    }
17    public static void print(){
18        if(head==null){
19            System.out.println(x:"Linked List is empty");
20        }
21        else{
22            Node1 temp=head;
23            while(temp!=null){
24                System.out.print(temp.data+" ");
25                temp=temp.next;
26            }
27        }
28    }
29    public static void MakeCycle(){
30        if(head==null){
```

Bottom Screenshot:

```
31        System.out.println(x:" ");
32    }
33    else{
34        Node1 temp=head;
35        Node1 n=null;
36        while(temp.next!=null){
37            n=temp;
38            temp=temp.next;
39        }
40        temp.next=n;
41    }
42    }
43    public static void CheckCycle(){
44        Node1 temp=head;
45        Node1 n=null;
46        int i;
47        for( i=1;i<size-1;i++){
48            n=temp;
49            temp=temp.next;
50            if(n.next==temp.next.next){
51                System.out.println(x:"Cycle exists");
52                break;
53            }
54        }if(i==size-1){
55            System.out.println(x:"Cycle does not exist");
56        }
57    }
58    public static void main(String[] args) {
59        Add(data:5);
```



THE END