Sukkur IBA University

Data Structure Algorithm

Name: Tariq Mehmood

CMS ID: 023-23-0127

Semester: 3rd

Section: E

Submitted by: Sir Riaz Hussain

Lab No: 04

Q1:

```
class Node{
    Node next;
    Node prev;
    int data;
    public Node (int data){
        this.data=data;
        this.next=null;
        this.prev=null;
    }
}
public class Task1{
    Node head;
    Node tail;
    int size=0;
    void addToFront(int data){
        size++;
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            tail=newNode;
        }
        else{
            newNode.next=head;
            head.prev=newNode;
            head=newNode;

        }
    }
    int getFrontItem(){
```

```
        if(head==null){
            System.out.println("Linked list is empty");
        }
        return head.data;

    }
    void removeFrontItem(){
        size--;
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        head=head.next;
    }
    void addToBack(int data){
        size++;
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            tail=newNode;
            return;
        }
        tail.next=newNode;
        newNode.prev=tail;
        tail=newNode;
    }
    int getBackItem(){

        if(head==null){
            System.out.println("Linked list is empty");
        }
```

```
        return tail.data;


    }
    void removeBackItem(){
        size--;
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        tail=tail.prev;
        tail.next=null;
    }
    void find(int key){
        Node temp=head;
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        while (temp!=null) {
            if(temp.data==key){
                System.out.println(key+"  is Presenet  in list");
                return;
            }
            temp=temp.next;
        }
        System.out.println(key+"  is not Presenet  in list");
    }
    void Remove(int key){
        size--;
        Node currnt=head;
        while(currnt!=null){
```

```
            if(currnt.data==key){
                if(currnt.prev!=null){
                    currnt.prev.next=currnt.next;
                }
                else{
                    head=currnt.next;
                }
                if(currnt.next!=null){
                    currnt.next.prev=currnt.prev;
                }
                else{
                    tail=currnt.prev;
                }
                return;
            }
            currnt=currnt.next;
        }
    }
    void addKeyBeforeNode(int key, int data){
        //size++;
        Node newNode=new Node(data);
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        Node temp=head;
        while (temp.next!=null) {



            temp=temp.next;
```

```
    }
}
void isListEmpty(){
    if(head==null){
        System.out.println("List is empty ");
        return;
    }
    System.out.println("The list has size is = "+size);

}
void addKeyAfterNode(int key, int data){
    //size++;
    Node newNode=new Node(data);
    if(head==null){
        System.out.println("Linked list is empty");
        return;
    }
    Node temp=head;
    while (temp.next!=null) {



        temp=temp.next;
    }
}
void display(){
   Node temp=head;
   if(head==null){
    System.out.println("Linked list is empty");
   }
   while (temp!=null) {
```

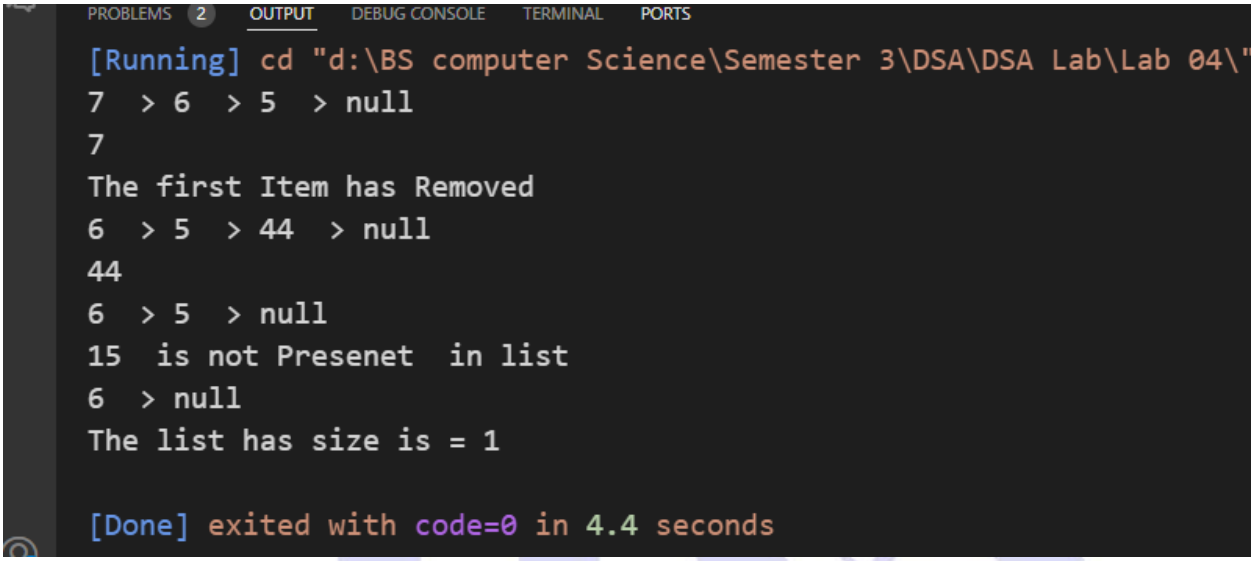
```
        System.out.print(temp.data+"  > ");
        temp=temp.next;
       }
       System.out.println("null");
  }
  public void Connect(){
      Node temp=head;
      while (temp!=null) {
       if(temp.next!=null){
            temp.next.prev=temp;
       }
       temp=temp.next;
      }
  }
  public static void main(String[] args) {
       Task1 DL=new Task1();
       DL.addToFront(5);
       DL.addToFront(6);
       DL.addToFront(7);
       DL.display();
       DL.addToBack(44);
       System.out.println(DL.getFrontItem());
       System.out.println("The first Item has Removed  ");
       DL.removeFrontItem();

       DL.display();
       System.out.println(DL.getBackItem());
       DL.removeBackItem();
       DL.display();
       DL.find(15);
       DL.Remove(5);
```

```
        //DL.addKeyBeforeNode(5,35);
        DL.display();
        DL.isListEmpty();

    }
}
```

Output



Q2:

```
class Node{
    Node next;
    Node prev;
    int data;
    public Node (int data){
        this.data=data;
        this.next=null;
        this.prev=null;
    }
}
public class Task2{
```

```
Node head;
Node tail;
int size=0;
void addToBack(int data){
    size++;
    Node newNode=new Node(data);
    if(head==null){
        head=newNode;
        tail=newNode;
        return;
    }
    tail.next=newNode;
    newNode.prev=tail;
    tail=newNode;
}
void removeBackItem(){
    size--;
    if(head==null){
        System.out.println("Linked list is empty");
        return;
    }
    tail=tail.prev;
    tail.next=null;
}
void PrintLnReverseOrder(){
    if(tail==null){
        System.out.println("Linked list is empty");
        return;
    }
    Node temp=tail;
    while (temp!=null) {
```

```
            System.out.print(temp.data+"   ");
            temp=temp.prev;

        } System.out.println("null");
    }
    void display(){
        Node temp=head;
        if(head==null){
         System.out.println("Linked list is empty");
        }
        while (temp!=null) {
         System.out.print(temp.data+"  > ");
         temp=temp.next;
        }
        System.out.println("null");
     }
    public static void main(String[] args) {
        Task2 DL=new Task2();
        DL.addToBack(5);
        DL.addToBack(6);
        DL.addToBack(7);
        DL.display();
        DL.PrintLnReverseOrder();


    }
}
```

Output

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 04\" && javac
5  > 6  > 7  > null
7   6   5   null

[Done] exited with code=0 in 1.943 seconds
```

Q3

```
class  Node {
    int data;
    Node next;
    Node (int data){
        this.data=data;
        this.next=null;
    }
}
public class Task3{
    Node head;
    int size=0;
    void addToFront(int data){
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            newNode.next=head;//Circular link
            return;
        }
        Node temp=head;
        while (temp.next!=head) {
            temp=temp.next;
        }
        temp.next=newNode;
        newNode.next=head;//newNode.next=head;//Circular Link
```

```
        head=newNode;
    }
    void addToBack(int data){
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            newNode.next=head;//Circular link
            return;
        }
        Node temp=head;
        while (temp.next!=head) {
            temp=temp.next;
        }
        temp.next=newNode;
        newNode.next=head;
    }
    void removeFront(){
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        head=head.next;
    }
    void removeBack() {
        if (head == null) {
           System.out.println("Linked list is empty");
           return;
        }
        if (head.next == head) { // Only one node
           head = null;
           return;
```

```
        }
        Node temp = head;
        while (temp.next.next != head) {
           temp = temp.next;
        }
        temp.next = head; // Second last node points to head
    }
    void display(){
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        Node temp=head;
        do {
            System.out.print(temp.data + " > ");
            temp = temp.next;
          } while (temp != head);
          System.out.println();
    }
    public static void main(String[] args) {
      Task3 CL=new Task3();
      CL.addToBack(45);
      CL.addToBack(6);
      CL.addToFront(76);
      CL.addToFront(46);
      CL.addToFront(56);
      CL.display();
      CL.removeFront();
      CL.removeBack();
      CL.display();
    }
```

}

Ouput

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 04\" &&
56 > 46 > 76 > 45 > 6 >
46 > 76 > 45 > 6 >

[Done] exited with code=0 in 1.911 seconds
```

Q4:

```
class  Node {
    int data;
    Node next;
    Node (int data){
        this.data=data;
        this.next=null;
    }
}
public class Task4{
    Node head;
    int size=0;
    void addToFront(int data){
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            newNode.next=head;//Circular link
            return;
        }
        Node temp=head;
        while (temp.next!=head) {
            temp=temp.next;
        }
        temp.next=newNode;
```

```
        newNode.next=head;//newNode.next=head;//Circular Link
        head=newNode;
    }
    void addToBack(int data){
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            newNode.next=head;//Circular link
            return;
        }
        Node temp=head;
        while (temp.next!=head) {
            temp=temp.next;
        }
        temp.next=newNode;
        newNode.next=head;
    }
    void removeFront(){
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        head=head.next;
    }
    boolean hasCycle(){
        if(head==null|| head.next==null){
            return false;
        }
        Node fast=head;
        Node slow=head;
        while (fast!=null && fast.next!=null ) {
```

```
            slow=slow.next;
            fast=fast.next.next;

            // If they meet, there's a cycle
            if (slow == fast) {
            return true;
            }
        }
        return false;
    }
    void display(){
        if(head==null){
            System.out.println("Linked list is empty");
            return;
        }
        Node temp=head;
        do {
            System.out.print(temp.data + " > ");
            temp = temp.next;
          } while (temp != head);
          System.out.println();
    }
    public static void main(String[] args) {
      Task4 CL=new Task4();
      CL.addToBack(45);
      CL.addToBack(6);
      CL.addToFront(76);
      CL.addToFront(46);
      CL.addToFront(56);
      CL.display();
      System.out.println("Does the list have a cycle? " + CL.hasCycle());
```

```
        //CL.removeFront();


        //CL.display();
        Task4 CL2=new Task4();
        Node a=new Node(1);
        Node b=new Node(2);
        Node c=new Node(3);
        Node d=new Node(4);
        a.next=b;
        b.next=c;
        c.next=d;
        d.next=null;
        CL2.head=a;
        System.out.println("Does the list have a cycle? " + CL2.hasCycle());



    }
}
```

Output

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 04\" &&
56 > 46 > 76 > 45 > 6 >
Does the list have a cycle? true
Does the list have a cycle? false

[Done] exited with code=0 in 1.915 seconds
```

Q5:

```
class  Node {
    int data;
    Node next;
    Node (int data){
        this.data=data;
```

```
        this.next=null;
    }
}
public class Task5{
Node head;
    int size=0;
    void addFront(int data){
        size++;
        Node newNode=new Node(data);
        if(head==null){
            head=newNode;
            return;
        }
        newNode.next=head;
        head=newNode;
    }
    void addToBack(int data){
        Node newNode=new Node(data);
        size++;
        Node temp=head;
        if(head==null){
            head=newNode;
            return;
        }
        else{
            while (temp.next!=null) {
                temp=temp.next;
            }
            temp.next=newNode;
        }
    }
```

```
int FindMiddleNode(){
    if(head==null){
        System.out.println("Linked list is empty");
    }
    Node fast=head;
    Node slow=head;
    while (fast!=null &&fast.next!=null ) {
        slow=slow.next;
        fast=fast.next.next;
    }
    System.out.println("The Middle Nodes Found  "+slow.data);
    return slow.data;
}
void display(){
    Node temp=head;
    if(head==null){
        System.out.println("Linked list is empty");

    }
    while (temp!=null) {
        System.out.print(temp.data+" >  ");
        temp=temp.next;
    }
    System.out.println("Null ");
}

public static void main(String[] args) {
    Task5 LinkedList=new Task5();
    LinkedList.addFront(4);
    LinkedList.addFront(5);
    LinkedList.addFront(6);
```

```
        LinkedList.addFront(7);
        LinkedList.addFront(8);
        LinkedList.addFront(9);
        LinkedList.display();
        System.out.println("Return Midddle Node   :"+LinkedList.FindMiddleNode());
    }
}
```

Output

```
[Running] cd "d:\BS computer Science\Semester 3\DSA\DSA Lab\Lab 04\" &&
9 >  8 >  7 >  6 >  5 >  4 >  Null
The Middle Nodes Found  6
Return Midddle Node   :6

[Done] exited with code=0 in 1.983 seconds
```

Sukkur
IBA

Sukkur
IBA

Sukkur
IBA