21BCE7371 RADHA KRISHNA GARG

DSA ASSIGNMENT -3

1.Write a Program to implement doubly linked list and its operations.

INPUT

```
import java.util.*;
public class que {
   class Node{
      Node next;
      int data;
      Node(int d){
         data=d;
      }
   }
   public static Node head;
   public Node front, rear;
   public que(){this.front=this.rear=null;}
   void push(int k){
      Node temp = new Node(k);
      if(this.rear==null){
         this.front= this.rear = temp;
         return;
      }
      this.rear.next=temp;
      this.rear=temp;
   }
   int pop(){
      if(this.front==null){
         return -1;
      }
      Node temp = this.front;
```

```
    this.front=this.front.next;
    if(this.front==null){
        this.rear=null;
    }
    return temp.data;
}
void prn(){
    Node t = this.front;
    while(t!=null){
        System.out.print(t.data+" -> ");
        t=t.next;
    }
    System.out.println();
}
public static void main(String args[]){
    Scanner sc = new Scanner(System.in);
    que q = new que();
    int f,v;
    do{
        System.out.println("========================================");
        System.out.println("----------SELECT AN OPTION:----------");
        System.out.println("========================================");
        System.out.println("1. TO PUSH TO QUEUE.");
        System.out.println("2. TO POP FROM QUEUE.");
        System.out.println("3. TO PRINT QUEUE.");
        System.out.println("4. TO EXIT!");
        System.out.println("========================================");
        System.out.println("Enter : ");
        f=sc.nextInt();
        System.out.println("========================================");
        switch(f){
```

```java
import java.util.*;
2 usages
public class que {
    7 usages
    class Node{
        3 usages
        Node next;
        3 usages
        int data;
        1 usage
        Node(int d){
            data=d;
        }
    }
    public static Node head;
    8 usages
    public Node front, rear;
    1 usage
    public que(){this.front=this.rear=null;}
    1 usage
    void push(int k){
        Node temp = new Node(k);
        if(this.rear==null){
            this.front= this.rear = temp;
            return;
        }
        this.rear.next=temp;
        this.rear=temp;
    }
    1 usage
    int pop(){
        if(this.front==null){
            return -1;
        }
        Node temp = this.front;
        this.front=this.front.next;
        if(this.front==null){
```

```
        if(this.front==null){
            this.rear=null;
        }
        return temp.data;
    }
    1 usage
    void prn(){
        Node t = this.front;
        while(t!=null){
            System.out.print(t.data+" -> ");
            t=t.next;
        }
        System.out.println();
    }
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        que q = new que();
        int f,v;
        do{
            System.out.println("========================================");
            System.out.println("-----------SELECT AN OPTION:------------");
            System.out.println("========================================");
            System.out.println("1. TO PUSH TO QUEUE.");
            System.out.println("2. TO POP FROM QUEUE.");
            System.out.println("3. TO PRINT QUEUE.");
            System.out.println("4. TO EXIT!");
            System.out.println("========================================");
            System.out.println("Enter : ");
            f=sc.nextInt();
            System.out.println("========================================");
            switch(f){
                case 1:
                    System.out.println("ENTER ELEMENT TO BE PUSHED: ");
                    v=sc.nextInt();
                    q.push(v);
                    break;
```

```
            System.out.println("==========================================");
            System.out.println("----------SELECT AN OPTION:-------------");
            System.out.println("==========================================");
            System.out.println("1. TO PUSH TO QUEUE.");
            System.out.println("2. TO POP FROM QUEUE.");
            System.out.println("3. TO PRINT QUEUE.");
            System.out.println("4. TO EXIT!");
            System.out.println("==========================================");
            System.out.println("Enter : ");
            f=sc.nextInt();
            System.out.println("==========================================");
            switch(f){
                case 1:
                    System.out.println("ENTER ELEMENT TO BE PUSHED: ");
                    v=sc.nextInt();
                    q.push(v);
                    break;
                case 2:
                    System.out.println("ELEMENT POPPED: "+q.pop());
                    break;
                case 3:
                    q.prn();
                    break;
                case 4:
                    break;
                default:
                    System.out.println("WRONG INPUT!");
            }
        }while(f!=4);
    }
}
```

```
==========================================
-----------SELECT AN OPTION:-------------
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
Enter :
1
==========================================
ENTER ELEMENT TO BE PUSHED:
23
==========================================
-----------SELECT AN OPTION:-------------
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
Enter :
1
==========================================
ENTER ELEMENT TO BE PUSHED:
45
==========================================
-----------SELECT AN OPTION:-------------
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
```

```
4. TO EXIT!
========================================
Enter :
1
========================================
ENTER ELEMENT TO BE PUSHED:
56
========================================
----------SELECT AN OPTION:------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
========================================
Enter :
178
========================================
WRONG INPUT!
========================================
----------SELECT AN OPTION:------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
========================================
Enter :
3
========================================
23 -> 45 -> 56 ->
========================================
----------SELECT AN OPTION:------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
```

```
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
Enter :
1
==========================================
ENTER ELEMENT TO BE PUSHED:
23
==========================================
-----------SELECT AN OPTION:-------------
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
Enter :
1
==========================================
ENTER ELEMENT TO BE PUSHED:
56
==========================================
-----------SELECT AN OPTION:-------------
==========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
==========================================
Enter :
1
==========================================
ENTER ELEMENT TO BE PUSHED:
```

```
34
========================================
-----------SELECT AN OPTION:-------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
========================================
Enter :
2
========================================
ELEMENT POPPED: 23
========================================
-----------SELECT AN OPTION:-------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
========================================
Enter :
34
========================================
WRONG INPUT!
========================================
-----------SELECT AN OPTION:-------------
========================================
1. TO PUSH TO QUEUE.
2. TO POP FROM QUEUE.
3. TO PRINT QUEUE.
4. TO EXIT!
========================================
Enter :
3
========================================
```

2.Write a Program to implement queue operations using linked list

INPUT

```
import java.util.Scanner;


public class dll {
   static Node head = null;
   class Node{
      int data;
      Node prev;
      Node next;
      Node(int d){data = d;}
   }
   public void push(int nD){
      Node nN = new Node(nD);
      nN.next = head;
      nN.prev = null;
      if(head!=null){
         head.prev = nN;
      }
      head = nN;
   }
   public void insB(Node nxN, int nd){
      if(nxN==null){
         System.out.println("The node can not be NULL");
      }
      Node nn = new Node(nd);
      nn.prev = nxN.prev;
      nxN.prev = nn;
      nn.next = nxN;
      if(nn.prev!=null){
         nn.prev.next = nn;
```

```
    }
    else{
        head=nn;
    }
}
public void insA(Node preN,int v){
    if(preN==null){
        System.out.println("The given node can not be null!");
    }
     Node newN = new Node(v);
     newN.prev = preN;
     newN.next = preN.next;
     preN.next = newN;
     if(newN.next==null){
         newN.next.prev = newN;
     }
}
public void append(int dat){
    Node newN = new Node(dat);
    Node last = head;
    newN.next = null;
    if(head==null){
        newN.prev=null;
        head = newN;
        return;
    }
    while(last.next!=null){
        last=last.next;
    }
    last.next=newN;
    newN.prev=last;
```

```
}
public static void disp(dll d){
   while (d.head!=null)
   {
      System.out.println(d.head.data+"");
      d.head = d.head.next;
   }
}
static void del(Node no){
   if(head==null||no==null){
      return;
   }
   if(head==no){
      head = no.next;
   }
   if(no.next!=null){
      no.next.prev=no.prev;
   }
   if (no.prev != null) {
      no.prev.next = no.next;
   }
   return;
}
static void deleteNodeAtGivenPos(int n)
{
   if (head == null || n <= 0)
      return;
   Node current = head;
   int i;
   for (i = 1; current != null && i < n; i++)
   {
```

```
        current = current.next;
    }
    if (current == null) {
        return;
    }
    del(current);
}
public static void main(String args[]){
    dll dl = new dll();
    int d, f=0;
    Scanner sc = new Scanner(System.in);
    while(f==0) {
        System.out.println("Input 1 to Input Nodes!");
        System.out.println("Input 2 to Delete Node!");
        System.out.println("Input 3 to EXIT.");
        System.out.println("====================================");
        int fl1 = sc.nextInt();
        switch (fl1) {
            case 1:
                System.out.println("Input 1 to Insert Data at the beginning.");
                System.out.println("Input 2 to Insert Data at the head");
                System.out.println("Input 3 to Insert Data at the end of the double linked list.");
                System.out.println("====================================");
                int fl2 = sc.nextInt();
                switch (fl2) {
                    case 1:
                        System.out.println("Input data to be pushed: ");
                        d = sc.nextInt();
                        dl.push(d);
                        break;
                    case 3:
```

```
                System.out.println("Input Data to be Appended: ");
                d = sc.nextInt();
                dl.append(d);
                break;
              default:
                System.out.println("WRONG INPUT!");
            }
            break;
          case 2:

        }
        disp(dl);
      }
    }
}
```

```java
import java.util.Scanner;

3 usages
public class dll {
    17 usages
    static Node head = null;
    16 usages
    class Node{
        2 usages
        int data;
        15 usages
        Node prev;
        19 usages
        Node next;
        4 usages
        Node(int d){data = d;}
    }
    1 usage
    public void push(int nD){
        Node nN = new Node(nD);
        nN.next = head;
        nN.prev = null;
        if(head!=null){
            head.prev = nN;
        }
        head = nN;
    }
    public void insB(Node nxN, int nd){
        if(nxN==null){
            System.out.println("The node can not be NULL");
```

```
        if(nxN==null){
            System.out.println("The node can not be NULL");
        }
        Node nn = new Node(nd);
        nn.prev = nxN.prev;
        nxN.prev = nn;
        nn.next = nxN;
        if(nn.prev!=null){
            nn.prev.next = nn;
        }
        else{
            head=nn;
        }
    }
    public void insA(Node preN,int v){
        if(preN==null){
            System.out.println("The given node can not be null!");
        }
         Node newN = new Node(v);
         newN.prev = preN;
         newN.next = preN.next;
         preN.next = newN;
         if(newN.next==null){
             newN.next.prev = newN;
         }
    }
    1 usage
    public void append(int dat){
        Node newN = new Node(dat);
        Node last = head;
        newN.next = null;
```

```
    Node newN = new Node(dat);
    Node last = head;
    newN.next = null;
    if(head==null){
        newN.prev=null;
        head = newN;
        return;
    }
    while(last.next!=null){
        last=last.next;
    }
    last.next=newN;
    newN.prev=last;
}
1 usage
public static void disp(dll d){
    while (d.head!=null)
    {
        System.out.println(d.head.data+"");
        d.head = d.head.next;
    }
}
1 usage
static void del(Node no){
    if(head==null||no==null){
        return;
    }
    if(head==no){
        head = no.next;
    }
```

```
        if(head==no){
            head = no.next;
        }
        if(no.next!=null){
            no.next.prev=no.prev;
        }
        if (no.prev != null) {
            no.prev.next = no.next;
        }
        return;
    }
    static void deleteNodeAtGivenPos(int n)
    {
        if (head == null || n <= 0)
            return;
        Node current = head;
        int i;
        for (i = 1; current != null && i < n; i++)
        {
            current = current.next;
        }
        if (current == null) {
            return;
        }
        del(current);
    }
    public static void main(String args[]){
        dll dl = new dll();
        int d, f=0;
```

```
        if (current == null) {
            return;
        }
        del(current);
    }
    public static void main(String args[]){
        dll dl = new dll();
        int d, f=0;
        Scanner sc = new Scanner(System.in);
        while(f==0) {
            System.out.println("Input 1 to Input Nodes!");
            System.out.println("Input 2 to Delete Node!");
            System.out.println("Input 3 to EXIT.");
            System.out.println("=====================================");
            int fl1 = sc.nextInt();
            switch (fl1) {
                case 1:
                    System.out.println("Input 1 to Insert Data at the beginning.");
                    System.out.println("Input 2 to Insert Data at the head");
                    System.out.println("Input 3 to Insert Data at the end of the double linked list.");
                    System.out.println("=====================================");
                    int fl2 = sc.nextInt();
                    switch (fl2) {
                        case 1:
                            System.out.println("Input data to be pushed: ");
                            d = sc.nextInt();
                            dl.push(d);
                            break;
                        case 3:
                            System.out.println("Input Data to be Appended: ");
```

```
                System.out.println("====================================");
                int fl2 = sc.nextInt();
                switch (fl2) {
                    case 1:
                        System.out.println("Input data to be pushed: ");
                        d = sc.nextInt();
                        dl.push(d);
                        break;
                    case 3:
                        System.out.println("Input Data to be Appended: ");
                        d = sc.nextInt();
                        dl.append(d);
                        break;
                    default:
                        System.out.println("WRONG INPUT!");
                }
                break;
            case 2:

        }
        disp(dl);
    }
}
}
```