Solution of Assignment 5

Implementation of Doubly Linked List (using java)

```
import java.util.*;

class Node
{
	protected int regd_no;
	protected float mark;
	protected Node next;
	protected Node prev;
}

public class doublyLinkedlist
{
	static Node start = null;
	static Node end = null;

	public static void Create()
	{
		Scanner sc=new Scanner(System.in);
		Node p;
		int ch;
		do
		{
			p=new Node();
			System.out.println("Enter registration number: ");
			p.regd_no=sc.nextInt();
			System.out.println("Enter marks number: ");
			p.mark=sc.nextInt();
			p.next=null;
			if(start == null)
			{
				start = p;
				end = p;
				p.prev = null;
			}
			else
```

```
                {
                        p.prev = end;
                        end.next=p;
                        end = p;
                }
                System.out.println("Do you want to create more number of nodes(y/n)");
                ch=sc.next().charAt(0);
        }while(ch=='y' || ch=='Y');
}

public static void Display()
{
        Node p=start;
        if(start == null)
        {
                System.out.println("empty linked list");
        }
        else
        {
                System.out.println("*******Node details********* \nReg.no --- marks");
                while(p!=null)
                {
                        System.out.println(p.regd_no+"---------"+p.mark);
                        p=p.next;
                }
        }
}

public static void InsBeg()
{
        Scanner sc=new Scanner(System.in);
        Node p=new Node();
        System.out.println("Enter registration number: ");
        p.regd_no=sc.nextInt();
        System.out.println("Enter marks number: ");
        p.mark=sc.nextInt();
        p.prev = null;
        if(start == null)
        {
                p.next = null;
                start = p;
                end = p;
        }
        else
```

```
        {
                start.prev = p;
                p.next = start;
                start = p;
        }
}

public static void InsEnd()
{
        Scanner sc=new Scanner(System.in);
        Node p=new Node();
        System.out.println("Enter registration number: ");
        p.regd_no=sc.nextInt();
        System.out.println("Enter marks number: ");
        p.mark=sc.nextInt();
        p.next = null;
        if(start == null)
        {
                p.prev = null;
                start = p;
                end = p;
        }
        else
        {
                 p.prev = end;
                 end.next=p;
                 end = p;
         }
}

public static void InsAny()
{
        Scanner sc = new Scanner(System.in);
        Node p = new Node();
        System.out.println("Enter the registration number of new node: ");
        p.regd_no = sc.nextInt();
        System.out.println("Enter the marks number of node: ");
        p.mark = sc.nextFloat();
        System.out.println("Enter position of new node: ");
         int pos = sc.nextInt();
         if(pos == 0)
         {
             System.out.println("Position does not exist in linked list: ");
         }
```

```
                else if(start == null)
                {
                        p.next = null;
                        p.prev = null;
                        start = p;
                        end = p;
                        System.out.println("Node add first position: ");
                }
                else if(pos == 1)
                {
                        p.next = start;
                        p.prev = null;
                        start.prev = p;
                        start = p;
                        System.out.println("Node add first position: ");
                }
                else
                {
                        Node q = start;
                        for(int i=1;i<pos-1 && q.next!=null;i++)
                        {
                                q = q.next;
                        }
                        if(q.next == null)
                        {
                                q.next = p;
                                p.next = null;
                                p.prev = q;
                                end = p;
                                System.out.println("position not found, so ne Node add last
position: ");
                        }
                        else
                        {
                                p.next = q.next;
                                p.prev = q;
                                q.next.prev = p;
                                q.next = p;
                                System.out.println("New node add "+pos+" position");
                        }
                }
        }
```

```
public static void DelBeg()
{
        if(start==null)
        {
                System.out.println("Empty linked list");
        }
        else if(start.next == null)
        {
                Node p = start;
                start = null;
                end = null;
                System.out.println("Delete node information\nReg.no --- marks");
                 System.out.println(p.regd_no+"---------"+p.mark);
        }
        else
        {
                 Node p = start;
                 start = start.next;
                 start.prev = null;
                 System.out.println("Delete node information\nReg.no --- marks");
                 System.out.println(p.regd_no+"---------"+p.mark);
        }
}

public static void DelEnd()
{
        if(start==null)
        {
                 System.out.println("Empty linked list");
        }
        else if(start.next == null)
        {
                Node p = start;
                start = null;
                end = null;
                System.out.println("Delete node information\nReg.no --- marks");
                System.out.println(p.regd_no+"---------"+p.mark);
        }
        else
        {
                Node p = end;
                end = end.prev;
                end.next = null;
                System.out.println("Delete node information\nReg.no --- marks");
```

```
                        System.out.println(p.regd_no+"---------"+p.mark);
                }
        }

        public static void DelAny()
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter position of deleted node: ");
                int pos = sc.nextInt();
                if(start == null)
                {
                        System.out.println("Empty linked list, delete not possible");
                }
                else if(pos==1 && start.next == null)
                {
                        Node q = start;
                        start =null;
                        end = null;
                        System.out.println("Deleted node info-- registration no: "+q.regd_no+" and
mark: "+q.mark);
                }
                else if(pos==1)
                {
                        Node q = start;
                        start = start.next;
                        start.prev = null;
                        System.out.println("Deleted node info-- registration no: "+q.regd_no+" and
mark: "+q.mark);
                }
                else
                {
                        Node q = start;
                        Node p = start;
                        for(int i=1;i<pos && p.next!=null;i++)
                        {
                                q = p;
                                p = p.next;
                        }
                        if(p.next == null)
                        {
                                System.out.println("position not found, delete not possible ");
                        }
                        else
                        {
```

```
                                System.out.println("Deleted node info-- registration no:
"+p.regd_no+" and mark: "+p.mark);
                                if(p.next == null)
                                {
                                        q.next = null;
                                        end = q;
                                 }
                                else
                                {
                                        p.next.prev = q;
                                        q.next = p.next;
                                }
                         }
                }
        }

        public static void Search(int regNo)
        {
                Scanner sc=new Scanner(System.in);
                if(start==null)
                {
                        System.out.println("Empty linked list");
                }
                else
                {
                        Node p = start;
                        while(p!=null)
                        {
                                if(p.regd_no == regNo)
                                {
                                        System.out.println("registration found, Enter the updated
marks");
                                        p.mark = sc.nextInt();
                                }
                                p = p.next;
                        }
                        System.out.println("Marks updated");
                }
        }

        public static void Count()
        {
                int c = 0;
                Node q = start;
```

```
        while(q!=null)
        {
                c++;
                q=q.next;
        }
        System.out.println("Number of nodes present in linked list is "+c);
}

public static void main(String[] args)
{
        Scanner sc=new Scanner(System.in);
         while(true)
         {
                System.out.println("\n****MENU*****");
                System.out.println("0:Exit");
                System.out.println("1:Creation");
                System.out.println("2:Display");
                System.out.println("3:Insert new node at the beginning");
                System.out.println("4:Insert new node at the end");
                System.out.println("5:Insert new node at any position");
                System.out.println("6:Delete a new node from first");
                System.out.println("7:Delete a new node from last");
                System.out.println("8:Delete a new node from any position");
                System.out.println("9:Update marks based on registration no.");
                System.out.println("10:Count of linked list");
                System.out.println("Enter the choice");
                int choice=sc.nextInt();
                switch(choice)
                {
                        case 0:System.exit(0);
                        case 1:  Create();      break;
                        case 2:  Display();     break;
                        case 3:  InsBeg();      break;
                        case 4:  InsEnd();      break;
                        case 5:  InsAny();      break;
                        case 6:  DelBeg();      break;
                        case 7:  DelEnd();      break;
                        case 8:  DelAny();      break;
                        case 9:  System.out.println("Enter registration no.");
                                 int regno = sc.nextInt();
                                 Search(regno);       break;
                        case 10:  Count();       break;

                        default:          System.out.println("Wrong choice");
```

```
                    }
                }
            }
        }
```