```java
import java.util.Scanner;

// Class to Represent a Node
	class Node
	{
	int info;
	Node next;
	}

	//  Class to Represent the Singly Linked List
	public class SinglyLinkedList
	{

	public static Node head=null;

	// Create a Singly Linked List
	public static void create()
	{
		Scanner sc=new Scanner (System.in);
		Node p=new Node();
		System.out.print("Input info ");
		p.info=sc.nextInt();
		p.next=null;
		head=p;

		//Adding Remaining Nodes (if any)
		System.out.println("Do you want more nodes(y/n)");
		char ch=sc.next().charAt(0);
		while(ch!='n')
		{
		Node q=new Node();
		System.out.print("Input info ");
		p.info=sc.nextInt();
		q.next=null;
		p.next=q;
		p=q;
		System.out.println("Do you want more nodes(y/n)");
		ch=sc.next().charAt(0);
		}

	}
	// Display the Content of the Singly Linked List
	public static void display()
	{
		Node temp = head;

		while(temp!=null)
```

```
        {
        System.out.print(temp.info+"--->");
        temp=temp.next;
        }
        System.out.print("null \n");

}

//Count the no. of nodes in the Singly Linked List
public static int count()
{
        int count=0;
        Node temp=head;
        while(temp!=null)
        {
        count = count + 1;
        temp=temp.next;
        }
        return count;
}

//Search a Particular Element of the Singly Linked List
public static int search(int x)
{
        int pos=0;
        Node temp=head;
        while(temp!=null)
        {
                pos++;
                if(temp.info==x)
                        return pos;
                temp=temp.next;
        }
        return -1;
}

// Insert at the Beginning of the Singly Linked List
public static void insert_beg()
{
        Scanner sc=new Scanner (System.in);
        Node p=new Node();
        System.out.print("Input info ");
        p.info=sc.nextInt();
        p.next=head;
        head=p;

}
```

```
	// Insert at the End of the Singly Linked List
	public static void insert_end()
	{
		Scanner sc=new Scanner (System.in);
		Node p=new Node();
		System.out.print("Input info ");
		p.info=sc.nextInt();
		if(head==null)
		{
		head=p;
		}
		else
		{
		Node temp=head;
		while(temp.next!=null)
		{
			temp=temp.next;
		}
		temp.next=p;
		}
	}

	// Insert at a Particular Position of the Singly Linked
List
	public static void insert_pos(int pos)
	{
		Scanner sc=new Scanner (System.in);
		int c=count();
		if(pos>=1&&pos<=c+1)
		{
			if(pos==1)
			{
				insert_beg();
			}
			else if(pos==c+1)
			{
				insert_end();
			}
			else
			{
				Node p=new Node();
				System.out.print("Input info ");
				p.info=sc.nextInt();
				Node temp=head;
				int cnt=1;
				while(cnt<pos-1)
```

```
                {
                cnt++;
                temp=temp.next;
                }
                p.next=temp.next;
                temp.next=p;

            }
        }
        else
        {
            System.out.println("Invalid Position");
        }
}

// Delete at the Beginning of the Singly Linked List
public static void delete_beg()
{
        if(head == null)
        {
        System.out.println("Underflow");
        return;
        }

        head=head.next;

}


// Delete at the End of the Singly Linked List
public static Node delete_end()
{
        Node temp=head;
        if(head==null)
        {
            System.out.println("Underflow");
            return null;
        }
        if(head.next==null)
        {
            head=null;

            return temp;
        }
        else
        {
```

```
            while(temp.next.next!=null)
            {
                  temp=temp.next;
            }
            temp.next=null;
            }
            return temp;
      }

      // Delete at a Particular Position of the Singly Linked
List
      public static void delete_pos()
      {

            if(head==null)
            {
                  System.out.println("Underflow");
                  return;
            }

            Scanner sc=new Scanner (System.in);
            System.out.println("Enter the position");
            int pos=sc.nextInt();
            int c=count();
            if(pos<=c)
            {
                  if(pos==1)
                  {
                        delete_beg();
                  }
                  else if(pos==c)
                  {
                        delete_end();
                  }
                  else
                  {

                        Node temp=head;
                        int cnt=1;
                        while(cnt<pos-1)
                        {
                        cnt++;
                        temp=temp.next;
                        }
                        temp.next=temp.next.next;

                  }
```

```
        }
        else
        {
              System.out.println("Invalid Position");
        }

}

// Reverse the Singly Linked List
public static void reverse()
{

        Node pvs = null;
        Node cur = head;
        Node nxt = null;


        while (cur != null)
        {
              nxt = cur.next;
              cur.next = pvs;
              pvs = cur;
              cur = nxt;
        }
        head = pvs;

}


// Sort the Singly Linked List
public static void sort()
{
    Node cur = head, index = null;

    if (head == null) {
        return;
    }
    else {
        while (cur != null) {

            index = cur.next;

            while (index != null) {

                if (cur.info > index.info) {
                    swap(cur,index);
                }
```

```
                index = index.next;
            }
            cur = cur.next;
        }
    }

}

// Swap the information of two nodes
public static void swap(Node cur,Node in)
{
      int temp_r;
      temp_r = cur.info;
    cur.info = in.info;
    in.info = temp_r;
}

public static void main(String[] args) {
     Scanner sc=new Scanner (System.in);

     while(true)
     {
     System.out.println("****MENU*****");
     System.out.println("0:Exit");
     System.out.println("1:Creation");
     System.out.println("2:Display");
     System.out.println("3:Count");
     System.out.println("4:Search");
     System.out.println("5:Insert");
     System.out.println("6:Delete");
     System.out.println("7:Reverse");
     System.out.println("8:Sort");
     System.out.println("*************");
     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:
```

```
                System.out.println("No. of Nodes ="+count());
                break;
            case 4:
                System.out.println("Enter info to be searched");
                int e=sc.nextInt();
                int pos=search(e);
                if(pos==-1)
                    System.out.println("Searched info not
present");
                else
                    System.out.println("Info present at position
"+pos);
                break;
            case 5:
                System.out.println("****INSERT*****");
                System.out.println("1: Begning");
                System.out.println("2: End");
                System.out.println("3: Specific Position");
                System.out.println("Enter the choice");
                int ch=sc.nextInt();
                switch(ch)
                {
                case 1:
                    insert_beg();
                    break;
                case 2:
                    insert_end();
                    break;
                case 3:
                    System.out.println("Enter the position");
                    pos=sc.nextInt();
                    insert_pos(pos);
                    break;
                default:
                    System.out.println("Wrong choice");
                    break;
                }
                break;
            case 6:
                System.out.println("****DELETE*****");
                System.out.println("1: Begning");
                System.out.println("2: End");
                System.out.println("3: Specific Position");
                System.out.println("Enter the choice");
                ch=sc.nextInt();
                switch(ch)
                {
```

```
                case 1:
                    delete_beg();
                    break;
                case 2:
                    Node t =delete_end();
                    System.out.println(t.info);
                    break;
                case 3:
                    delete_pos();
                    break;
                default:
                    System.out.println("Wrong choice");
                    break;
                }
                break;
            case 7:
                reverse();
                break;
            case 8:
                sort();
                break;
            default:
            System.out.println("Wrong choice");
            }
            }
    }

}
```