Assignment 11

Title: Interface and Packages.

Problem Statement: Write a JAVA program which demonstrates concept of interfaces and packages. Use of customized interfaces and packages is expected.

Objective:
To understand use of interfaces and packages.

Outcome:
To be able to understand and implement interfaces in JAVA. To use packages for different applications.

Theory:
Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, no body).

- Interfaces specify what a class must do and not how. It is the blueprint of the class.
- An Interface is about capabilities like a Player may be an interface and any class implementing Player must be able to (or must implement) move(). So it specifies a set of methods that the class has to implement.
- If a class implements an interface and does not provide method bodies for all functions specified in the interface, then the class must be declared abstract.
- A Java library example is, Comparator Interface. If a class implements this interface, then it can be used to sort a collection.

Code:
Declaration:

```
package Declaration;
public interface List
{
        void create();
        void display();
```

```
        int count();
        void add_at_beg(int x);
        void add_at_end(int x);
        void add_at_posn(int x,int pos);
        void del_at_beg();
        void del_at_end();
        void del_at_posn(int pos);
}

Definition:
package Definition;
import Declaration.List;
import java.util.*;
class Node
{
        int data;
        Node next;

        Node(int x)
        {
                data = x;
                next = null;
        }

        public void finalize()
        {
                System.out.println("Node deleted.");
        }
}
class MyList implements List
{
        Node head;

        MyList()
        {
                head = null;
        }

        public void create()
        {
```

```
        String str;
        Scanner sc = new Scanner(System.in);
        Node p = null;
        while(true)
        {
                System.out.println("Enter data: ");
                str = sc.next();
                if(str.equals("stop"))
                {
                        break;
                }
                try
                {
                        int x = Integer.parseInt(str);
                        if(head == null)
                        {
                                head = new Node(x);
                                p = head;
                        }
                        else
                        {
                                p.next = new Node(x);
                                p = p.next;
                        }
                }
                catch(NumberFormatException e)
                {
                        System.out.println(e);
                }
        }
}
public void display()
{
        Node p = head;
        while(p != null)
        {
                System.out.println(p.data);
                p = p.next;
        }
}
```

```
public int count()
{
	Node p = head;
	int cnt = 0;
	while(p != null)
	{
		cnt++;
		p = p.next;
	}
	return cnt;
}
public void add_at_beg(int x)
{
	Node p = new Node(x);
	p.next = head;
	head = p;
}
public void add_at_end(int x)
{
	if(head == null)
	{
		add_at_beg(x);
	}
	else
	{
		Node p = head;
		while(p.next != null)
		{
			p = p.next;
		}
		p.next = new Node(x);
	}
}
public void add_at_posn(int x,int pos)
{
	int cnt = count();
	if(pos < 1 || pos > cnt+1)
	{
		System.out.println("Invalid position.");
		return;
```

```
        }
        if(pos == 1)
        {
                add_at_beg(x);
        }
        else
        {
                Node p = head;
                Node q = null;
                for(int i=0;i<pos-1;i++)
                {
                        q = p;
                        p = p.next;
                }
                Node r = new Node(x);
                r.next = p;
                q.next = r;
        }
}
public void del_at_beg()
{
        if(head != null)
        {
                Node p = head;
                head = p.next;
                p = null;
                System.gc();
                try
                {
                        Thread.sleep(25);
                }
                catch(InterruptedException e)
                {
                        System.out.println(e);
                }
        }
        else
        {
                System.out.println("List empty.");
        }
```

```
}
public void del_at_end()
{
	if(head != null)
	{
		Node p = head;
		Node q = null;
		while(p.next != null)
		{
			q = p;
			p = p.next;
		}
		if(q != null)
		{
			q.next = null;
			p = null;
		}
		else
		{
			p = null;
			head = null;
		}
		System.gc();
		try
		{
			Thread.sleep(25);
		}
		catch(InterruptedException e)
		{
			System.out.println(e);
		}
	}
	else
	{
		System.out.println("List empty.");
	}
}
public void del_at_posn(int pos)
{
	int cnt = count();
```

```
            if(pos < 1 || pos > cnt)
            {
                System.out.println("Invalid position.");
                return;
            }
            if(pos == 1)
            {
                del_at_beg();
            }
            else
            {
                Node p = head;
                Node q = null;
                for(int i=0;i<pos-1;i++)
                {
                    q = p;
                    p = p.next;
                }
                q.next = p.next;
                p = null;
                System.gc();
                try
                {
                    Thread.sleep(25);
                }
                catch(InterruptedException e)
                {
                    System.out.println(e);
                }
            }
        }
}
class MyClass
{
        public static void main(String[] args)
        {
            MyList obj = new MyList();
            Scanner sc = new Scanner(System.in);
            int choice,data,pos;
            while(true)
```

```
{
        try
        {
                System.out.println("1. Create list.");
                System.out.println("2. Display list.");
                System.out.println("3. Add at Beginning.");
                System.out.println("4. Add at End.");
                System.out.println("5. Add at Position.");
                System.out.println("6. Delete at Beginning.");
                System.out.println("7. Delete at End.");
                System.out.println("8. Delete at Position.");
                System.out.println("9. Exit.");
                System.out.println("Enter choice: ");
                choice = sc.nextInt();
                switch(choice)
                {
                        case 1:
                                obj.create();
                                break;
                        case 2:
                                obj.display();
                                break;
                        case 3:
                                System.out.println("Enter data to add: ");
                                data = sc.nextInt();
                                obj.add_at_beg(data);
                                break;
                        case 4:
                                System.out.println("Enter data to add: ");
                                data = sc.nextInt();
                                obj.add_at_end(data);
                                break;
                        case 5:
                                System.out.println("Enter data to add: ");
                                data = sc.nextInt();
                                System.out.println("Enter position: ");
                                pos = sc.nextInt();
                                obj.add_at_posn(data,pos);
                                break;
                        case 6:
```

```
                                        obj.del_at_beg();
                                        break;
                                case 7:
                                        obj.del_at_end();
                                        break;
                                case 8:
                                        System.out.println("Enter position: ");
                                        pos = sc.nextInt();
                                        obj.del_at_posn(pos);
                                        break;
                                case 9:
                                        return;
                                default:
                                        System.out.println("Invalid input.");
                        }
                }
                catch(InputMismatchException e)
                {
                        System.out.println(e);
                        sc.next();
                }
        }
    }
}
```

Test case:

1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.

Enter choice:
1
Enter data:
1
Enter data:

2
Enter data:
3
Enter data:
4
Enter data:
5
Enter data:
stop
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
2
1
2
3
4
5
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
3
Enter data to add:
6
1. Create list.
2. Display list.

3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
2
6
1
2
3
4
5
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
4
Enter data to add:
7
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
5
Enter data to add:

8
Enter position:
3
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
2
6
1
8
2
3
4
5
7
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
6
Node deleted.
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.

7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
7
Node deleted.
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
8
Enter position:
3
Node deleted.
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.
5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
2
1
2
3
4
5
1. Create list.
2. Display list.
3. Add at Beginning.
4. Add at End.

5. Add at Position.
6. Delete at Beginning.
7. Delete at End.
8. Delete at Position.
9. Exit.
Enter choice:
9

Conclusion:
We learnt packages and interfaces and successfully implemented the same.