*****************Assignment – 6 – STATE DESIGN PATTERN************************

********************** SMD PACKAGE ******************

1.)Course.java

```
package smd;

import java.util.ArrayList;
import java.util.Date;
import java.util.GregorianCalendar;

public class Course {
	private ArrayList<Student> registeredStudents = new ArrayList<Student>();
	private State state;
	private Date date;

	public State getState() {
		return state;
	}

	public void setState(State state) {
		this.state = state;
	}

	public Date getDate() {
		return date;
	}

	public void setDate(Date date) {
		this.date = date;
	}

	public ArrayList<Student> getRegisteredStudents() {

		return registeredStudents;
	}

	public void addStudent(Student s)
	{
		this.isWithinRange(s);

		Open open = new Open();
		if(open.getClass().equals(this.state.getClass()))
		{
			this.registeredStudents.add(s);
		}

	}

	public void isWithinRange(Student s)
	{
```

```
		GregorianCalendar g = new GregorianCalendar(2016,5,15);
		Date proposedDate = g.getTime();

		if(s.getRegDate().before(proposedDate))
		{
			this.setState(new Proposed());
			System.out.println("\n\t Since course is in proposed state,student " +
s.getRollNo() +  " cannot be registered.");
		}
		else if(s.getRegDate().after(proposedDate) || s.getRegDate().equals(proposedDate) )
		{
			g = new GregorianCalendar(2016,5,30);
			Date proposedEndDate = g.getTime();

			if(s.getRegDate().before(proposedEndDate) ||
s.getRegDate().equals(proposedEndDate))
			{
				this.setState(new Open());
				if(this.registeredStudents.size() < 4 &&
(s.getRegDate().equals(proposedEndDate)))
				{
					System.out.println("\n\t Course is cancelled.");
					this.setState(new Cancelled());

				}
				else if(this.registeredStudents.size() >= 10)
				{
					System.out.println("\n\t Course is closed,student " +
s.getRollNo() +  " could not be registered.No of registrations full.");
					this.setState(new Closed());
				}
				else
				{
					System.out.println("\n\t Since course registration is
open,student " + s.getRollNo() +  " is registered.");
				}
			}
			else
			{
				if(this.registeredStudents.size() < 4)
				{
					System.out.println("\n\t Course is cancelled.");
					this.setState(new Cancelled());
				}
				else
				{
					System.out.println("\n\t Course is closed,student " +
s.getRollNo() +  " could not be registered.Post end date registrations not allowed.");
					this.setState(new Closed());
				}
			}
```

```
		}
	}

	public void displayState()
	{
		this.getState().display();
	}
}
```

2.)State.java

```
package smd;


public abstract class State {

	public abstract void display();
}
```

3.)Student.java

```
package smd;

import java.util.Date;

public class Student {
	private int rollNo;
	private Date regDate;

	public Student(int rollNo, Date regDate) {
		super();
		this.rollNo = rollNo;
		this.regDate = regDate;
	}

	public int getRollNo() {
		return rollNo;
	}

	public void setRollNo(int rollNo) {
		this.rollNo = rollNo;
	}

	public Date getRegDate() {
		return regDate;
	}

	public void setRegDate(Date regDate) {
		this.regDate = regDate;
	}
```

```
}
```

4.)Proposed.java

```
package smd;

public class Proposed extends State{

	@Override
	public void display() {
		// TODO Auto-generated method stub
		System.out.println("State:Proposed");
	}

}
```

5.)Open.java

```
package smd;

public class Open  extends State{

	@Override
	public void display() {
		// TODO Auto-generated method stub
		System.out.println("State:Open");
	}

}
```

6.)Cancelled.java

```
package smd;

public class Cancelled  extends State{

	@Override
	public void display() {
		// TODO Auto-generated method stub
		System.out.println("State:Cancelled");
	}

}
```

7.)Closed.java

```
package smd;

public class Closed  extends State{

	@Override
	public void display() {
		// TODO Auto-generated method stub
```

```
			System.out.println("State:Closed");
		}

}
```

8.)TestClient.java

**********CLIENT PACKAGE**********

```
package Client;

import java.util.Date;
import java.util.GregorianCalendar;

import smd.Course;
import smd.Student;


public class TestClient {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		GregorianCalendar g = new GregorianCalendar(2016,0,1);
		Date d1 = g.getTime();

		g = new GregorianCalendar(2016,5,20);
		Date d2 = g.getTime();

		g = new GregorianCalendar(2016,5,15);
		Date d3 = g.getTime();
		g = new GregorianCalendar(2016,5,20);
		Date d4 = g.getTime();
		g = new GregorianCalendar(2016,5,22);
		Date d5 = g.getTime();
		g = new GregorianCalendar(2016,5,25);
		Date d6 = g.getTime();
		g = new GregorianCalendar(2016,5,30);
		Date d7 = g.getTime();

		g = new GregorianCalendar(2016,5,17);
		Date d8 = g.getTime();



		g = new GregorianCalendar(2016,5,18);
		Date d9 = g.getTime();

		g = new GregorianCalendar(2016,5,17);
		Date d10 = g.getTime();
```

```
		g = new GregorianCalendar(2016,5,16);
		Date d11 = g.getTime();
		g = new GregorianCalendar(2016,5,21);
		Date d12 = g.getTime();
		g = new GregorianCalendar(2016,6,16);
		Date d13 = g.getTime();

		Course course = new Course();


		Student s1 = new Student(1,d1);
		Student s2 = new Student(2,d2);
		Student s3 = new Student(3,d3);
		Student s4 = new Student(4,d4);
		Student s5 = new Student(5,d5);
		Student s6 = new Student(6,d6);
		Student s7 = new Student(7,d7);
		Student s8 = new Student(8,d8);
		Student s9 = new Student(9,d9);
		Student s10 = new Student(10,d10);
		Student s11 = new Student(11,d11);
		Student s12 = new Student(12,d12);
		Student s13 = new Student(13,d13);

		course.addStudent(s1);
		course.addStudent(s2);
		course.addStudent(s3);
		course.addStudent(s4);
		course.addStudent(s5);
		course.addStudent(s6);
		course.addStudent(s7);
		course.addStudent(s8);
		course.addStudent(s9);
		course.addStudent(s10);
		course.addStudent(s11);
		course.addStudent(s12);
		course.addStudent(s13);

		System.out.println("\n\t Total number of students registered for the course : " +
course.getRegisteredStudents().size());
	}

}
```

5.) Output

Since course is in proposed state,student 1 cannot be registered.

Since course registration is open,student 2 is registered.

Since course registration is open,student 3 is registered.

Since course registration is open,student 4 is registered.

Since course registration is open,student 5 is registered.

Since course registration is open,student 6 is registered.

Since course registration is open,student 7 is registered.

Since course registration is open,student 8 is registered.

Since course registration is open,student 9 is registered.

Since course registration is open,student 10 is registered.

Since course registration is open,student 11 is registered.

Course is closed,student 12 could not be registered.No of registrations full.

Course is closed,student 13 could not be registered.Post end date registrations not allowed.

Total number of students registered for the course : 10
*/