# Department of Computing

**SE-210: Software Design and Architecture**

**Class:** BESE-9AB

# Lab 12: Behavioral Design Patterns

# Instructor: Dr. Hasan Ali Khattak

# 

# Ahmed Hassan Ismail – BESE-9B – 237897

# Lab 12: Behavioral Design Patterns

### Introduction:

Students will have hands-on experience of implementing a behavioral design pattern to a given problem.

### Lab Objectives:

This objective of this lab is to get a practical understanding and knowledge of the *Observer Design Pattern*. After the completion of this lab, students will be able to apply observer pattern to a given scenario.

### Helping Material:

Please consult lecture slides on LMS.

### Tools:

Papyrus

### Lab Tasks

### Task 1

Assume that you have to create a weather application where a user inputs current temperature as a float value. You have a Subject class as a concrete class which will read the input. It will have five different functions: attach, detach, setState, getState, notification. Upon calling setState(float temperature), the value of the local variable temp will be set with the temperature and the notification() function will be called. The notification() function will notify all attached observers by calling their respective update methods. You have six different observers: DataRecorder, ForecastGenerator, DataAnalyzer, GUIUpdater, NewsGenerator, GraphsUpdater. The observers will attach themselves with the subject through parameterized constructor. In the update method of each observer, you have to read current temperature by calling getState() method and print a meaningful string on the console e.g. for DataRecorder you can write: ‘Temperature ’+ currentTemperature + ‘ has been written in local database’.

Use following client code for testing:

public static void main( String[] args ) {

Subject sub = new Subject();

new DataRecorder (sub);

new ForecastGenerator (sub);

new DataAnalyzer (sub);

new GUIUpdater (sub);

new NewsGenerator(sub);

new GraphsUpdater(sub);

Scanner scan = new Scanner(System.in);

for (int i = 0; i < 10; i++) {

System.out.print("\nEnter a number: ");

sub.setState(scan. nextFloat());

}

}

Based on the scenario given above, your tasks are as follows:

1. Draw UML Class diagram of the solution – after applying observer design pattern.
2. Fully functional code of the solution. Use the client code given above.
3. Bonus Question: Make sure only one observer can attach itself to the Subject (What about using Singleton design pattern here?)
4. You must be able to demonstrate your code in next lab.

**Answer:**

|  |
| --- |
| Solution |
| UML Class Diagram  sdfghjk  Screenshots of the output:    Source Code: Zip your source code and upload it as well. |

### Deliverables

Compile a single word document by filling in the solution part and submit this Word file on LMS. This lab grading policy is as follows: The lab is graded between 0 to 10 marks. The submitted solution can get a maximum of 5 marks. At the end of each lab or in the next lab, there will be a viva related to the tasks. The viva has a weightage of 5 marks. Insert the solution/answer in this document. You must show the implementation of the tasks in the designing tool, along with your completed Word document to get your work graded. You must also submit this Word document on the LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to **Sundas Dawood** <sundas.dawood@seecs.edu.pk>