**The Observer Design Pattern**

**Problem:**

The objective of this exercise is to implement the Observer design pattern.

**Observer pattern steps:**

1. Create an interface called **AlarmListener**. This is the observer interface. In **AlarmListener**, there is a void **alarm**() method defined.
2. Create a class called **SensorSystem**. This is the publisher class. Define one instance variable

ArrayList<AlarmListener > listeners = new ArrayList();

This ArrayList saves all the observers of this publisher.

1. In **SensorSystem**, define a method void **register**(AlarmListener alarmListener). What it does is to add alarmListener to the ArrayList listeners.
2. In **SensorSystem**, define another method void **soundTheAlarm**(). What it does is to use a for loop to loop through all the listeners/observers in the ArrayList listeners, and call their alarm() method.
3. Different three concrete observer classes: **Lighting**, **Gates**,and **Surveillance**. Make them implement the interface **AlarmListener.** Implement the **alarm**() method in all three classes**.** In **Lighting**, **alarm**() prints out “lights up”. In **Gates**, **alarm**() prints out “gates close”. In **Surveillance**, **alarm**() prints out “Surveillance – by the numbers:”.
4. Use the following client code to try it.

**public** **class** **ObservserDemo** {

**public** **static** **void** main( **String**[] args ) {

SensorSystem sensorSystem = **new** SensorSystem();

sensorSystem.register(**new** Gates());

sensorSystem.register(**new** Lighting());

sensorSystem.register(**new** Surveillance());

sensorSystem.soundTheAlarm();

}

}

Upload your code to the Blackboard when you are done.