**Exercise 7: Implementing the Observer Pattern**

**Observer.java**

package com.mycompany.observerpattern;

public interface Observer {

    void update(String stockName, double stockPrice);

}

**MobileApp.java**

package com.mycompany.observerpattern;

public class MobileApp implements Observer {

    private String user;

    public MobileApp(String user) {

        this.user = user;

    }

    @Override

    public void update(String stockName, double stockPrice) {

        System.out.println(" MobileApp [" + user + "] - " + stockName + " updated to ₹" + stockPrice);

    }

}

**Stock.java**

package com.mycompany.observerpattern;

public interface Stock {

    void registerObserver(String stockName, Observer o);

    void removeObserver(String stockName, Observer o);

    void notifyObservers(String stockName, double price);

}

**StockMarket.java**

package com.mycompany.observerpattern;

import java.util.\*;

public class StockMarket implements Stock {

    private Map<String, Double> stockPrices = new HashMap<>();

    private Map<String, List<Observer>> stockObservers = new HashMap<>();

    public void setStockPrice(String stockName, double price) {

        stockPrices.put(stockName, price);

        System.out.println("\n[Update] " + stockName + " = ₹" + price);

        notifyObservers(stockName, price);

    }

    @Override

    public void registerObserver(String stockName, Observer o) {

        stockObservers.putIfAbsent(stockName, new ArrayList<>());

        stockObservers.get(stockName).add(o);

    }

    @Override

    public void removeObserver(String stockName, Observer o) {

        if (stockObservers.containsKey(stockName)) {

            stockObservers.get(stockName).remove(o);

        }

    }

    @Override

    public void notifyObservers(String stockName, double price) {

        if (stockObservers.containsKey(stockName)) {

            for (Observer o : stockObservers.get(stockName)) {

                o.update(stockName, price);

            }

        }

    }

}

**WebApp.java**

package com.mycompany.observerpattern;

public class WebApp implements Observer {

    private String user;

    public WebApp(String user) {

        this.user = user;

    }

    @Override

    public void update(String stockName, double stockPrice) {

        System.out.println(" WebApp [" + user + "] - " + stockName + " updated to ₹" + stockPrice);

    }

}

**ObserverPattern.java**

package com.mycompany.observerpattern;

public class ObserverPattern {

    public static void main(String[] args) {

        StockMarket market = new StockMarket();

        Observer mobileAlice = new MobileApp("Alice");

        Observer webBoris = new WebApp("Boris");

        Observer mobileCyrus = new MobileApp("Cyrus");

        market.registerObserver("TCS", mobileAlice);

        market.registerObserver("TCS", webBoris);

        market.registerObserver("INFY", mobileCyrus);

        market.registerObserver("WIPRO", webBoris);

        market.registerObserver("WIPRO", mobileAlice);

        market.setStockPrice("TCS", 3500);

        market.setStockPrice("INFY", 1480.5);

        market.setStockPrice("WIPRO", 410.75);

        market.removeObserver("WIPRO", mobileAlice);

        market.setStockPrice("WIPRO", 420.00);

    }

}

**Output**

