**Exercise 7: Implementing the Observer Pattern**

import java.util.\*;

interface Stock {

void registerObserver(Observer o);

void removeObserver(Observer o);

void notifyObservers();

}

interface Observer {

void update(String stockSymbol, double stockPrice);

}

class StockMarket implements Stock {

private List<Observer> observers;

private String stockSymbol;

private double stockPrice;

public StockMarket() {

observers = new ArrayList<>();

}

@Override

public void registerObserver(Observer o) {

observers.add(o);

}

@Override

public void removeObserver(Observer o) {

observers.remove(o);

}

@Override

public void notifyObservers() {

for (Observer observer : observers) {

observer.update(stockSymbol, stockPrice);

}

}

public void setStockPrice(String symbol, double price) {

this.stockSymbol = symbol;

this.stockPrice = price;

notifyObservers();

}

}

class MobileApp implements Observer {

private String appName;

public MobileApp(String appName) {

this.appName = appName;

}

@Override

public void update(String stockSymbol, double stockPrice) {

System.out.println(appName + " received update: " + stockSymbol + " is now $" + stockPrice);

}

}

class WebApp implements Observer {

private String siteName;

public WebApp(String siteName) {

this.siteName = siteName;

}

@Override

public void update(String stockSymbol, double stockPrice) {

System.out.println(siteName + " displays: " + stockSymbol + " at $" + stockPrice);

}

}

public class ObserverPatternExample {

public static void main(String[] args) {

StockMarket stockMarket = new StockMarket();

Observer mobileApp = new MobileApp("StockMobile");

Observer webApp = new WebApp("StockWeb");

stockMarket.registerObserver(mobileApp);

stockMarket.registerObserver(webApp);

stockMarket.setStockPrice("AAPL", 150.25);

stockMarket.setStockPrice("GOOGL", 2750.00);

stockMarket.removeObserver(webApp);

stockMarket.setStockPrice("TSLA", 695.50);

}

}

