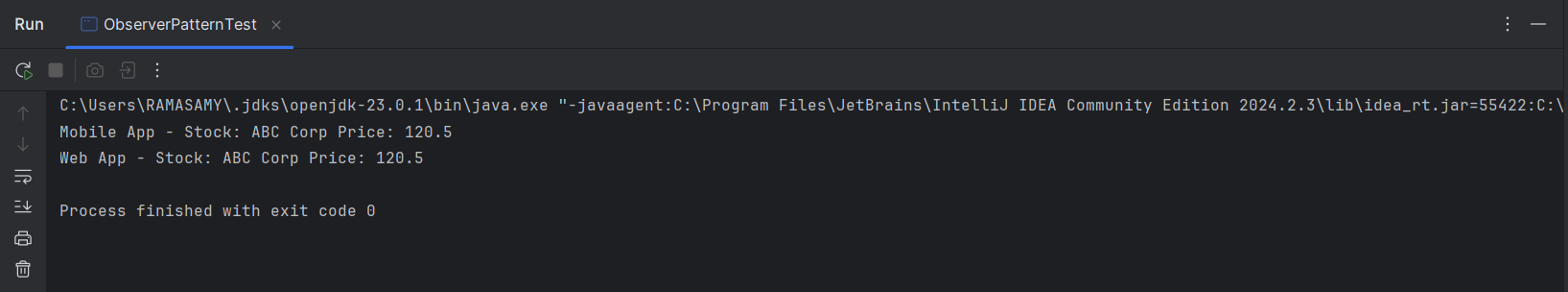
**Exercise 7: Implementing the Observer Pattern**

**CODE:**

**ObserverPatternTest.java**

import java.util.\*;  
  
interface Observer {  
 void update(String stockName, double price);  
}  
  
interface Stock {  
 void registerObserver(Observer o);  
 void removeObserver(Observer o);  
 void notifyObservers();  
}  
  
class StockMarket implements Stock {  
 private List<Observer> observers = new ArrayList<>();  
 private String stockName;  
 private double price;  
  
 public void setStockPrice(String name, double price) {  
 this.stockName = name;  
 this.price = price;  
 notifyObservers();  
 }  
  
 public void registerObserver(Observer o) {  
 observers.add(o);  
 }  
  
 public void removeObserver(Observer o) {  
 observers.remove(o);  
 }  
  
 public void notifyObservers() {  
 for (Observer o : observers) {  
 o.update(stockName, price);  
 }  
 }  
}  
  
class MobileApp implements Observer {  
 public void update(String stockName, double price) {  
 System.*out*.println("Mobile App - Stock: " + stockName + " Price: " + price);  
 }  
}  
  
class WebApp implements Observer {  
 public void update(String stockName, double price) {  
 System.*out*.println("Web App - Stock: " + stockName + " Price: " + price);  
 }  
}  
  
class ObserverPatternTest {  
 public static void main(String[] args) {  
 StockMarket stockMarket = new StockMarket();  
 Observer mobile = new MobileApp();  
 Observer web = new WebApp();  
  
 stockMarket.registerObserver(mobile);  
 stockMarket.registerObserver(web);  
  
 stockMarket.setStockPrice("ABC Corp", 120.5);  
 }  
}

**OUTPUT:**

****