МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Кафедра інформаційних систем

та мереж

**Звіт**  до лабораторної роботи №6

з дисципліни “Екстремальне програмування ”

**Тестування коду програми**

Виконав: студент гр. КН-311 Туз О.І.

Прийняв:

Щербак С.С.

Львів – 2020

Теоретичні відомості

Тестування програмного коду - процес виконання програмного коду, спрямований на виявлення існуючих в ньому дефектів. Під дефектом тут розуміється ділянку програмного коду, виконання якого за певних умов призводить до несподіваного поведінки системи (тобто поведінки, що не відповідає вимогам). Несподіване поведінку системи може призводити до збоїв у її роботі і відмов, в цьому випадку говорять про істотні дефектах програмного коду. Деякі дефекти викликають незначні проблеми, що не порушують процес функціонування системи, але кілька утрудняють роботу з нею. У цьому випадку говорять про середні або малозначних дефектах.

Завдання тестування при такому підході - визначення умов, при яких виявляються дефекти системи, і протоколювання цих умов. У завдання тестування зазвичай не входить виявлення конкретних дефектних ділянок програмного коду і ніколи не входить виправлення дефектів - це завдання налагодження, яка виконується за результатами тестування системи.

Мета застосування процедури тестування програмного коду - мінімізація кількості дефектів (особливо істотних) в кінцевому продукті.Тестування саме по собі не може гарантувати повної відсутності дефектів у програмному коді системи. Проте, у поєднанні з процесами верифікації та валідації, спрямованими на усунення суперечливості і неповноти проектної документації (зокрема - вимог на систему), грамотно організоване тестування дає гарантію того, що система задовольняє вимогам і веде себе відповідно з ними у всіх передбачених ситуаціях.

Найбільш популярні бібліотеки — JUnit і TestNg, і в даній лабораторній роботі використано першу. Вона є простим і гнучким фреймворком для тестування.

JUnit — [бібліотека](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0) для [тестування програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) для мови [Java](https://uk.wikipedia.org/wiki/Java). Створений [Кентом Беком](https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D0%BD%D1%82_%D0%91%D0%B5%D0%BA) і [Еріком Гаммою](https://uk.wikipedia.org/w/index.php?title=%D0%95%D1%80%D1%96%D0%BA_%D0%93%D0%B0%D0%BC%D0%BC%D0%B0&action=edit&redlink=1), JUnit є представником родини фреймворків [xUnit](https://uk.wikipedia.org/w/index.php?title=XUnit&action=edit&redlink=1) для різних мов програмування, яка бере початок у [SUnit](https://uk.wikipedia.org/w/index.php?title=SUnit&action=edit&redlink=1) Кента Бека для [Smalltalk](https://uk.wikipedia.org/wiki/Smalltalk). JUnit породив екосистему різноманітних розширень.

Досвід одержаний при роботі з JUnit був важливим у розробці концепцій [тестування програмного забезпечення](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

Виконання лабораторної роботи

Код класу який тестується

**package** model;  
  
**import** service.AdditionalInfo;  
  
**import** java.sql.Date;  
  
**public class** Order {  
 **private** User **user**;  
 **private** Date **date**;  
 **private int personAmount**;  
 **private boolean specialOccasion**;  
 **private boolean separateRoom**;  
 **private** Hall **hall**;  
 **private** String **additionalAdvise**;  
  
 **public** Order(User user, String date, **int** personAmount, String specialOccasion, String separateRoom, Hall hall, String additionalAdvise) {  
 **this**.**user** = user;  
 **this**.**date** = Date.*valueOf*(date);  
 **this**.**personAmount** = personAmount;  
 **this**.**specialOccasion** = (specialOccasion.equals(**"Yes"**)) ? **true** : **false**;  
 **this**.**separateRoom** = (separateRoom.equals(**"Yes"**)) ? **true** : **false**;  
 **this**.**hall** = hall;  
 **this**.**additionalAdvise** = additionalAdvise;  
 }  
  
 **public** User getUser() {  
 **return user**;  
 }  
  
 **public** Date getDate() {  
 **return date**;  
 }  
  
 **public int** getPersonAmount() {  
 **return personAmount**;  
 }  
  
 **public boolean** isSpecialOccasion() {  
 **return specialOccasion**;  
 }  
  
 **public boolean** isSeparateRoom() {  
 **return separateRoom**;  
 }  
  
 **public** Hall getHall() {  
 **return hall**;  
 }  
  
 **public** String getAdditionalAdvise() {  
 **return additionalAdvise**;  
 }  
  
 **public boolean** checkFreeDate() {  
 **if** (AdditionalInfo.*getBusyDates*().contains(**date**)) **return false**;  
 **else return true**;  
 }  
  
 **public double** calculatePrice() {  
 **boolean** isVipUser = **false**;  
 **for** (User vipUser : AdditionalInfo.*getListOfVipUsers*()) {  
 **if** (vipUser.getPhone().equals(**user**.getPhone())) {  
 isVipUser = **true**;  
 **break**;  
 }  
 }  
 **double** initialPrice = **hall**.getPrice();  
 **if** (**separateRoom**) initialPrice \*= 3; *//окрема кімната втричі дорожча* **if** (**personAmount** >= 5) initialPrice -= (**personAmount** / 5) \* 10; *//за кожного 5-го відвібувача знижка 10$* **if** (isVipUser) {  
 initialPrice \*= 0.9; *//vip клієнти мають 10% знижки* } **else** {  
 **if** (**specialOccasion**) initialPrice \*= 0.98; *//при святі ресторан дарує не vip клієнтам знжку 2%* }  
 **return** initialPrice;  
 }  
}

Код тестування OrderTest.java

**package** test;  
  
**import** model.Hall;  
**import** model.Order;  
**import** model.User;  
**import** org.junit.Assert;  
**import** org.junit.Test;  
  
**public class** OrderTest {  
 User **simpleUser** = **new** User(**"Pavlo"**, **"Rolson"**, **"pavlo.rolson@gmail.com"**, **"0991234567"**);  
 User **vipUser** = **new** User(**"Ivan"**, **"Petrenko"**, **"ivan.petrenko@gmail.com"**, **"0971111111"**);  
  
 Order **order** = **new** Order(**simpleUser**, **"2020-05-06"**, Integer.*valueOf*(10), **"Yes"**, **"Yes"**, Hall.***TERRACE***, **"no addition"**);  
 Order **vipOrder** = **new** Order(**vipUser**, **"2020-05-07"**, Integer.*valueOf*(10), **"Yes"**, **"Yes"**, Hall.***TERRACE***, **"no addition"**);  
  
 @Test  
 **public void** testCalculatePrice() {  
 Assert.*assertTrue*(**order**.calculatePrice() == 333.2);  
 Assert.*assertTrue*(**vipOrder**.calculatePrice() == 306.0);  
 }  
  
 @Test  
 **public void** testCheckFreeDate() {  
 Assert.*assertTrue*(**order**.checkFreeDate() == **false**);  
 Assert.*assertTrue*(**vipOrder**.checkFreeDate() == **true**);  
 }  
}

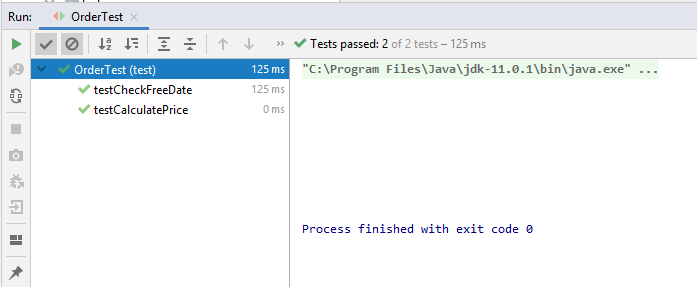


Рис.1 Результат виконання тестів

У класі Order.java побудована логіка оформлення замовлення і підрахунку його суми. Також в ньому перевіряється чи є замовник віп користувачем (враховується при обрахунку суми) і чи вільна дата замовлення.

У класі OrderTest.java введені різноманітні тестові дані: віп користувач і звичайний, вільна і зайнята дата замовлення. Усі ці данні передаються у якості параметрів у тест і проводиться перевірка очікуваного і даного результату. При умові що ці дані збігаються тест проходить успішно. Усе це потрібно для перевірки правильності роботи програми при введені різних параметрів.

Висновок: у даній лабораторній роботі було ознайомлено з базовими поняттями тестування коду, а також реалізація частини коду тестування веб-сайту за допомогою бібліотеки JUnit.