Лабораторная работа №7
Регрессионное тестирование

Цель работы: научиться производить регрессионное тестирование.
Отчет по лабораторной работе: тестируемая программа, отчет о проведении регрессионного тестирования.
Задание: Разработать регрессионные тесты для программы, разрабатывавшейся и тестировавшейся в лабораторных работах №3,4,5. Внести изменения в программу (переработать структуру классов, добавить новую функциональность, исправить ошибки). Провести регрессионное тестирование. Составить отчет о результате проведенного регрессионного тестирования.

Отчет о проведении регрессионного тестирования

1. Введение

1.1. Цель регрессионного тестирования

Целью регрессионного тестирования является проверка существующего функционала системы после внесения изменений, чтобы убедиться, что новые изменения не вызвали ошибок в уже работающих частях программы.

1.2. Область тестирования

Регрессионное тестирование охватывает:

- Проверку функциональности классов PhoneStore, ServiceCenter, Smartphone, AndroidSmartphone, IOSSmartphone.
- Проверку новой функциональности, добавленной в программу.
- Проверку исправленных ошибок.

2. Изменения в программе

В систему была добавлена новая функция для класса PhoneStore, позволяющая фильтровать смартфоны по операционной системе (OS). Теперь метод getSmartphonesByOS(String os) возвращает список смартфонов, соответствующих указанной операционной системе.

```
 1. package ru.miet.CourceTesting.Lr7;
 2.
 3. import java.util.List;
 4.
 5. import ru.miet.CourceTesting.Lr3.Smartphone;
 6. import ru.miet.CourceTesting.Lr4.PhoneStore;
 7.
 8. public class PhoneStoreEx extends PhoneStore {
 9.
10.     public PhoneStoreEx(String storeName) {
11.       super(storeName);
12.       // TODO Auto-generated constructor stub
13.     }
14.
15.     List<Smartphone> getSmartphonesByOS(String os) {
16.       return smartphones.stream().filter(x -> x.getOs() == os).toList();
17.     }
18. }
```

3. Регрессионные тесты

3.1. Тестовые сценарии

3.1.1. Проверка существующей функциональности

Все исходные тесты были разделены на подмножества:

- Множество тестов, пригодных для повторного использования (Все исходные тесты)
- Множество тестов, требующих повторного запуска (Выявлено не было)
- Множество устаревших тестов (Выявлено не было)
- Новые тесты (**testFeatures**).

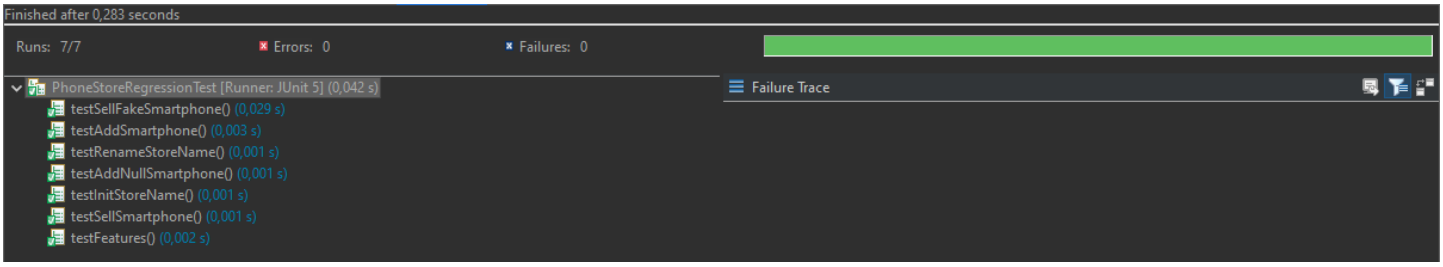
3.1.2. Проверка новой функциональности

1. **testFeatures**: Фильтрация смартфонов по операционной системе.
    - **Входные данные**: Запрос на получение списка смартфонов с os = "ios".
    - **Ожидаемый результат**: Возвращается список только с IOSSmartphone.

```
 1. package ru.miet.CourceTesting.Lr7;
 2.
 3. import static org.junit.jupiter.api.Assertions.assertEquals;
 4.
 5. import org.junit.jupiter.api.BeforeEach;
 6. import org.junit.jupiter.api.Test;
 7.
 8. import ru.miet.CourceTesting.Lr4.AndroidSmartphone;
 9. import ru.miet.CourceTesting.Lr4.IOSSmartphone;
10. import ru.miet.CourceTesting.Lr4.PhoneStoreTest;
11.
12. class PhoneStoreRegressionTest extends PhoneStoreTest {
13.
14.     @Override
15.     @BeforeEach
16.     public void setUp() {
17.   store = new PhoneStoreEx("Тестовый Магазин");
18.     }
19.
20.     @Test
21.     void testFeatures() {
22.   var smartphone = new AndroidSmartphone("Samsung", "Galaxy S21", 6.2, 0.7, "11", true);
23.   var smartphone1 = new IOSSmartphone("Apple", "iPhone 13", 6.1, 0.7, "15", true);
24.
25.   store.addSmartphone(smartphone);
26.   store.addSmartphone(smartphone1);
27.
28.   var ex = (PhoneStoreEx) store;
29.   assertEquals(1, ex.getSmartphonesByOS("iOS").size());
30.     }
31. }
```

3.2. Результаты регрессионного тестирования

Все тестовые сценарии изменённого модуля прошли успешно:



4. Заключение

Регрессионное тестирование подтвердило, что внесенные изменения не повлияли на существующую функциональность системы. Новая функция фильтрации смартфонов по операционной системе работает корректно, а ранее исправленные ошибки больше не возникают.