**Тестирование программного продукта**

Тестирование проводилось на уровне модуля методом «белого ящика» над классом DBHandler, так как данный класс напрямую взаимодействует с базой данных.

Для тестирования были выбраны такие методы класса, как: addContract(), updateContract(), deleteContract(), getAll(), getIncompleted(), getCompleted(), isContractCorrect().

Для проведения тестирования были составлены тест-кейсы и тестовый граф.

|  |  |
| --- | --- |
| Тест-кейс №: | 1 |
| Наименование: | Добавление данных договора |
| Предусловие: | Добавить данные договора в базу данных. |
| Постусловие: | После добавления программа переходит в первоначальное состояние. |
| Тестовая среда: | IntelliJ IDEA, JUnit5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Шаг | Вводимые данные | Ожидаемый результат | Фактический результат |
| 1 | Ввод данных договора | "2000-01-01", 15, "test.docx", "", (byte) 0; | "2000-01-01", 15, "test.docx", "", (byte) 0; | "2000-01-01", 15, "test.docx", "", (byte) 0; |
| 2 | Добавление данных договора | "2000-01-01", 15, "test.docx", "", (byte) 0; | Данные договора записываются в базу данных | Данные договора записаны в базу данных |
| 3 | Получение данных договора из базы данных |  | Данные договора из базы данных присваиваются объекту | Данные договора из базы данных присвоены объекту |
| 4 | Сравнение введенных данных с добавленными | Введенные данные;  Добавленные данные | Данные совпадают | Данные совпадают |

|  |  |
| --- | --- |
| Тест-кейс №: | 2 |
| Наименование: | Изменение данных договора |
| Предусловие: | Изменить данные договора в базе данных. |
| Постусловие: | После изменения программа переходит в первоначальное состояние. |
| Тестовая среда: | IntelliJ IDEA, JUnit5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Шаг | Вводимые данные | Ожидаемый результат | Фактический результат |
| 1 | Получение данных договора |  | Данные договора из базы данных присваиваются объекту | Данные договора из базы данных присвоены объекту |
| 2 | Изменение данных договора | "newTest.docx", "Description", (byte) 1, "2001-01-01"; | "2000-01-01", 15, "newTest.docx", "Description", (byte) 1, "2001-01-01"; | "2000-01-01", 15, "newTest.docx", "Description", (byte) 1, "2001-01-01"; |
| 3 | Обновление данных договора в базе данных | "2000-01-01", 15, "newTest.docx", "Description", (byte) 1, "2001-01-01"; | Данные договора в базе данных перезаписываются | Данные договора в базе данных перезаписаны |
| 4 | Получение данных перезаписанного договора из базы данных |  | Данные перезаписанного договора из базы данных присваиваются объекту | Данные перезаписанного договора из базы данных присвоены объекту |
| 5 | Сравнение измененных данных с перезаписанными | Измененные данные договора;  Полученные данные договора | Данные совпадают | Данные совпадают |

|  |  |
| --- | --- |
| Тест-кейс №: | 3 |
| Наименование: | Вывод данных договоров |
| Предусловие: | Вывести данные договоров из базы данных |
| Постусловие: | После вывода программа переходит в первоначальное состояние. |
| Тестовая среда: | IntelliJ IDEA, JUnit5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Шаг | Вводимые данные | Ожидаемый результат | Фактический результат |
| 1 | Ввод списка данных договоров | "2000-01-01", 1, "test.docx", "", (byte) 0;  "2000-01-01", 2, "test.docx", "", (byte) 0; | "2000-01-01", 1, "test.docx", "", (byte) 0;  "2000-01-01", 2, "test.docx", "", (byte) 0; | "2000-01-01", 1, "test.docx", "", (byte) 0;  "2000-01-01", 2, "test.docx", "", (byte) 0; |
| 2 | Добавление списка данных договоров | "2000-01-01", 1, "test.docx", "", (byte) 0;  "2000-01-01", 2, "test.docx", "", (byte) 0; | Список данных договоров записывается в базу данных | Список данных договоров записан в базу данных |
| 3 | Получение списка всех записей данных договоров из базы данных |  | Записи данных всех договоров из базы данных присваиваются объекту | Записи данных всех договоров из базы данных присвоены объекту |
| 4 | Получение списка записей данных неисполненных договоров из базы данных |  | Записи данных неисполненных договоров из базы данных присваиваются объекту | Записи данных неисполненных договоров из базы данных присвоены объекту |
| 5 | Получение списка записей данных исполненных договоров из базы данных |  | Записи данных исполненных договоров из базы данных присваиваются объекту | Записи данных исполненных договоров из базы данных присвоены объекту |
| 6 | Сравнение размеров полученных списков | Список данных всех документов;  Список данных неисполненных документов;  Список данных исполненных документов | 3;  2;  1; | 3;  2;  1; |

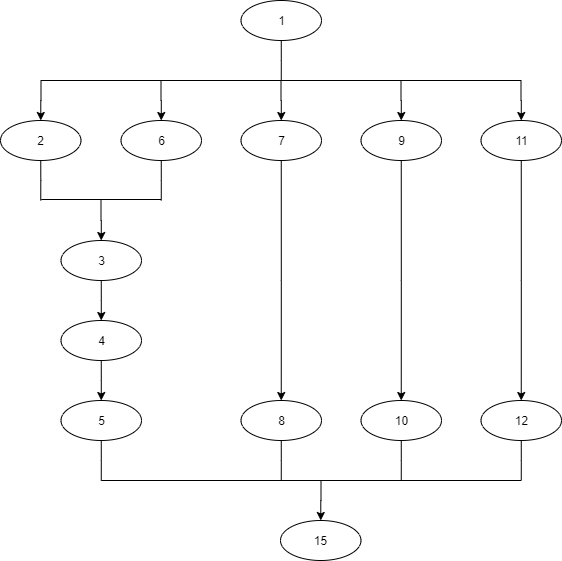
|  |  |
| --- | --- |
| Тест-кейс №: | 4 |
| Наименование: | Проверка существования договора |
| Предусловие: | Проверить существования указанного договора в базе данных. |
| Постусловие: | После проверки программа переходит в первоначальное состояние. |
| Тестовая среда: | IntelliJ IDEA, JUnit5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Шаг | Вводимые данные | Ожидаемый результат | Фактический результат |
| 1 | Ввод идентификатора существующего договора в базе данных | 2 | true | true |
| 2 | Ввод идентификатора несуществующего договора в базе данных | 5 | false | false |

|  |  |
| --- | --- |
| Тест-кейс №: | 5 |
| Наименование: | Удаление данных договора |
| Предусловие: | Удалить данные договора из базы данных. |
| Постусловие: | После удаления программа переходит в первоначальное состояние. |
| Тестовая среда: | IntelliJ IDEA, JUnit5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Шаг | Вводимые данные | Ожидаемый результат | Фактический результат |
| 1 | Указание идентификатора удаляемых данных договора | 15;  1;  2; | 15;  1;  2; | 15;  1;  2; |
| 2 | Удаление данных договоров с указанными идентификаторами из базы данных |  | Данные договоров с указанными идентификаторами удаляются из базы данных | Данные договоров с указанными идентификаторами удалены из базы данных |
| 3 | Получение списка записей данных договоров из базы данных |  | Записи данных договоров из базы данных присваиваются объекту | Записи данных договоров из базы данных присвоены объекту |
| 4 | Сравнение размера полученного списка | Полученный список | 0; | 0; |

Тестовый граф:



1. Запуск
2. Добавление неисполненного договора
3. Получение списка всех договоров
4. Получение данных последнего добавленного договора
5. Проверка данных последнего добавленного договора
6. Изменение статуса исполнения и даты исполнения договора
7. Добавление дополнительных договоров
8. Проверка размера всех видов списков
9. Проверка на существование номера существующего договора
10. Проверка на существование номера несуществующего договора
11. Удаление добавленных договоров
12. Проверка размера списка всех договоров
13. Завершение

**Tests.java**

package sample;

import javafx.collections.ObservableList;

import org.junit.jupiter.api.MethodOrderer.OrderAnnotation;

import org.junit.jupiter.api.Order;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import static org.junit.jupiter.api.Assertions.assertTrue;

import static org.junit.jupiter.api.Assertions.assertFalse;

import org.junit.jupiter.api.TestMethodOrder;

import sample.tables.\*;

@TestMethodOrder(OrderAnnotation.class)

class Tests {

private DBHandler db = new DBHandler();

private Contract testContract;

private Contract testingContract;

private ObservableList contracts;

@Test

@Order(1)

void addContractTest() {

testContract = new Contract("2000-01-01", 15, "test.docx", "", (byte) 0, null);

db.addContract(testContract);

contracts = db.getAll();

testingContract = (Contract) contracts.get(contracts.size() - 1);

assertEquals("01.01.2000", testingContract.getDate());

assertEquals(testContract.getNumber(), testingContract.getNumber());

assertEquals(testContract.getPath(), testingContract.getPath());

assertEquals(testContract.getDescription(), testingContract.getDescription());

assertEquals(testContract.getStatus(), testingContract.getStatus());

assertEquals(testContract.getCompletionDate(), testingContract.getCompletionDate());

}

@Test

@Order(2)

void updateContractTest() {

contracts = db.getAll();

testContract = (Contract) contracts.get(contracts.size() - 1);

testContract.setPath("newTest.docx");

testContract.setDescription("Description");

testContract.setStatus((byte) 1);

testContract.setCompletionDate("2001-01-01");

db.updateContract(testContract);

contracts = db.getAll();

testingContract = (Contract) contracts.get(contracts.size() - 1);

assertEquals(testContract.getPath(), testingContract.getPath());

assertEquals(testContract.getDescription(), testingContract.getDescription());

assertEquals(testContract.getStatus(), testingContract.getStatus());

}

@Test

@Order(3)

void getMethodsTest() {

db.addContract(new Contract("2000-01-01", 1, "test.docx", "", (byte) 0, null));

db.addContract(new Contract("2000-01-01", 2, "test.docx", "", (byte) 0, null));

assertEquals(3, db.getAll().size());

assertEquals(2, db.getIncompleted().size());

assertEquals(1, db.getCompleted().size());

}

@Test

@Order(4)

void isContractExistTest() {

assertTrue(db.isContractExist(2));

assertFalse(db.isContractExist(5));

}

@Test

@Order(5)

void deleteContractTest() {

db.deleteContract(1);

db.deleteContract(2);

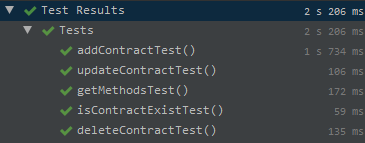
db.deleteContract(15);

assertEquals(0, db.getAll().size());

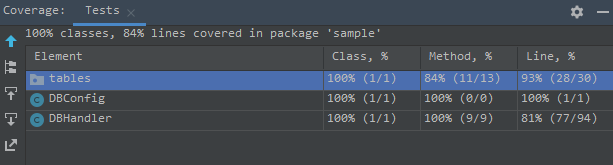
}

}

Результаты:



Статистика покрытия тестами:



Все результаты тестирования совпали с ожидаемыми.