МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
“Национальный исследовательский университет ИТМО”

**ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ
И КОМПЬЮТЕРНОЙ ТЕХНИКИ**

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине
“Тестирование программного обеспечения”

Вариант: 1000.

Работу выполнил:
— Студент группы P3311
Болорболд Аригуун
Лектор:
Клименков Сергей Викторович
Практик:
Птицын Максим Евгеньевич

г. Санкт-Петербург
2025 г.

Содержимое

1 Текст задания

1. Для указанной функции провести модульное тестирование разложения функции в степенной ряд. Выбрать достаточное тестовое покрытие.
2. Провести модульное тестирование указанного алгоритма. Для этого выбрать характерные точки внутри алгоритма, и для предложенных самостоятельно наборов исходных данных записать последовательность попадания в характерные точки. Сравнить последовательность попадания с эталонной.
3. Сформировать доменную модель для заданного текста. Разработать тестовое покрытие для данной доменной модели.

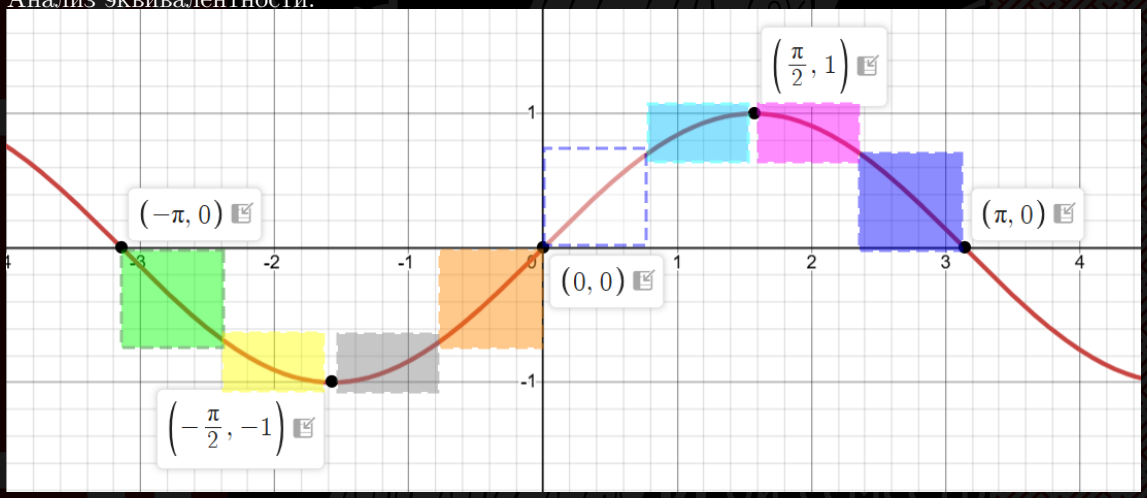
1. Функция `sin(x)`
2. Программный модуль для сортировки массива `методом вставок` http://www.cs.usfca.edu/ galles/visualization/ComparisonSort.html
3. Описание предметной области:
   Зажужжал мотор. Тоненький свист перерос в рев воздуха, вырывающегося в черную пустоту, усеянную невероятно яркими светящимися точками. Форд и Артур вылетели в открытый космос, как конфетти из хлопушки. Глава 8

2 Выполнение

Сслыка на репозиторию: GitHub

2.1 Тригонометрическая функция

Анализ эквивалентности:



Один из методов теста:

lab1/TrigTest.java

```
@ParameterizedTest(name = "sin({0})")
@DisplayName("Trig Function Test 1: Zero Values")
@ValueSource(doubles = {-PI, -0.5 * PI, 0, 0.5 * PI, PI})
void checkPiValues(double param) {
    assertAll(
            () -> assertEquals(Math.sin(param),
            sine.calculate(param, 69), 0.001)
    );
}
```

2.2 Алгоритм

Исходный код алгоритма (сортировка слиянием):

lab1/algo/sorts/SortAlgorithm.java

```
package ru.itmo.cs.kdot.lab1.algo.sorts;

import java.util.List;

public interface SortAlgorithm {
    int[] sort(int[] array);

    <K extends Comparable<? super K>, V> List<MapEntry<K, V>>
    sort(List<MapEntry<K, V>> map);
}
```

lab1/algo/sorts/comparisons/InsertionSort.java

```
package ru.itmo.cs.kdot.lab1.algo.sorts.comparisons;

import ru.itmo.cs.kdot.lab1.algo.sorts.MapEntry;
import ru.itmo.cs.kdot.lab1.algo.sorts.SortAlgorithm;

import java.util.List;

public class InsertionSort implements SortAlgorithm {
    @Override
    public int[] sort(int[] array) {
        if (array == null || array.length == 0){
            throw new IllegalArgumentException("Зачем хочешь
            сортировать пустой массив, лошара?)");
        }
        for(int i = 1; i < array.length; i++) {
            int key = array[i];
            int j = i - 1;

            while(j >= 0 && array[j] > key){
                array[j + 1] = array[j];
                j = j - 1;
            }
            array[j + 1] = key;
        }
        return array;
    }

    public <K extends Comparable<? super K>, V> List<MapEntry<K, V>> sort(List<MapEntry<K, V>> map) {
        if (map == null || map.isEmpty()) {
            throw new IllegalArgumentException("Зачем хочешь
            сортировать пустую словарь, лошара?)");
        }
        for(int i = 1; i < map.size(); i++) {
            MapEntry<K, V> index = map.get(i);
            int j = i - 1;

            while(j >= 0 && map.get(j).getKey().compareTo(index.getKey()) > 0) {
                map.set(j + 1, map.get(j));
                j = j - 1;
            }
```

```
            map.set(j + 1, index);
        }
        return map;
    }
}
```

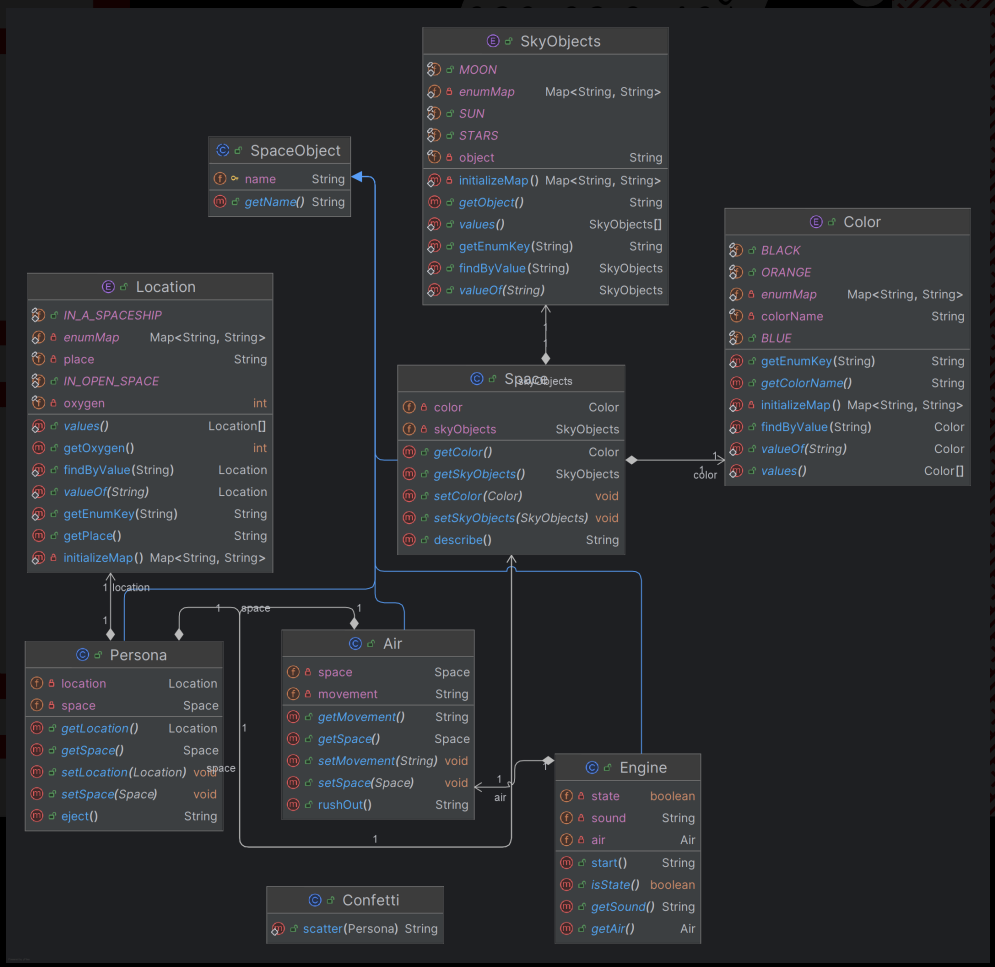
Один из методов теста:

lab1/algo/SortTest.java

```
    @Test
    @DisplayName("Test for a random array consisting of non-
    positive random hardcoded values")
    void checkRandomNonPositiveHardcodedSorting() {
        assertAll(
                () -> assertArrayEquals(new int[]{-97, -80, -66, -51, -46, -38, -35, -34, -17, -6},
                insertionSort.sort(new int[]{-97, -80, -51, -6, -17, -34, -46, -66, -38, -35})),
                () -> assertArrayEquals(new int[]{-297, -256, -243, -230, -228, -207, -205, -199, -19
                insertionSort.sort(new int[]{-297, -197, -228, -230, -199, -20, -45, -256, -10, -146,
                () -> assertArrayEquals(new int[]{-299, -296, -285, -284, -260, -250, -245, -227, -22
                insertionSort.sort(new int[]{-121, -21, -20, -284, -260, -250, -245, -299, -227, -223
        );
    }
```

2.3 Предметная область

UML:



Один из методов теста:

lab1/DomainTest.java

```
@Test
@DisplayName("Domain Test 3: Engine")
void checkEngine(){
    assertAll(
            () -> assertEquals("Мотор", engine.getName()),
            () -> assertFalse(engine.isState()),
            () -> assertEquals("затих", engine.getSound())
    );
    engine.start();
    assertAll(
            () -> assertEquals("Мотор", engine.getName()),
            () -> assertTrue(engine.isState()),
            () -> assertEquals("зажужжал", engine.getSound()),
            () -> assertEquals("Мотор зажужжал", engine.start())
    );
}
```

3 Дополнительное задание

Настроить автоматические тесты в Github Workflow.

3.1 Реализация

.github/workflows/build-and-test.yml

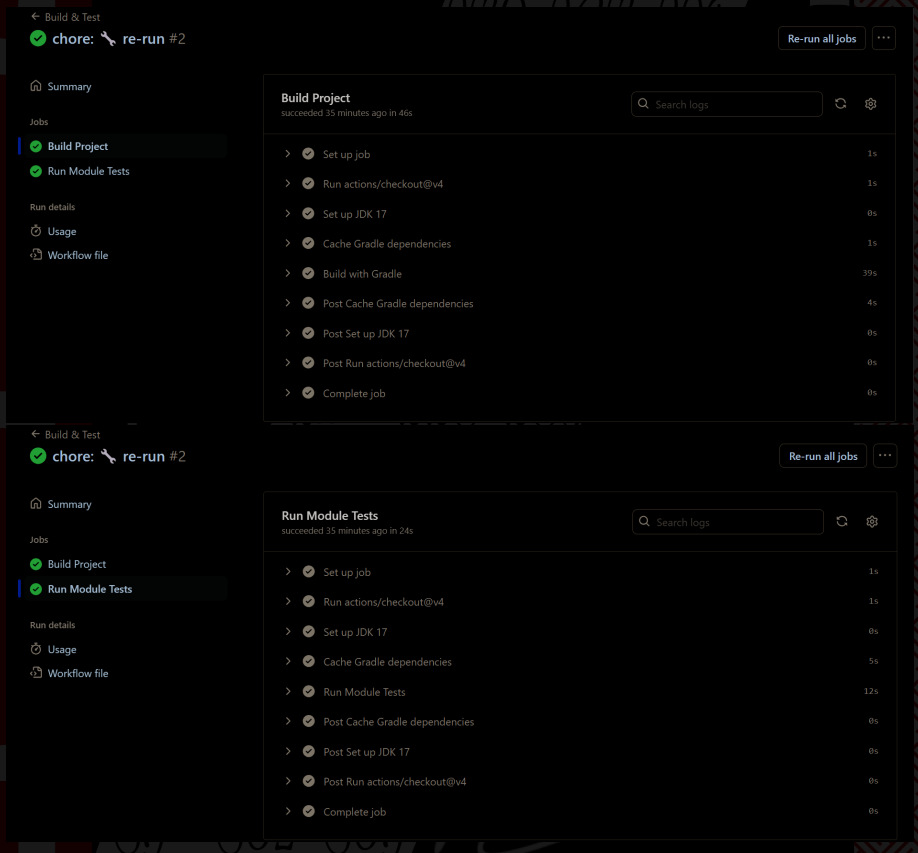
```
name: Build & Test

on:
  push:
    branches: [main]

jobs:
  build:
    name: Build Project
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - name: Set up JDK 17
        uses: actions/setup-java@v4
        with:
          java-version: 17
          distribution: 'temurin'
      - name: Cache Gradle dependencies
        uses: actions/cache@v3
        with:
          path: |
            ~/.gradle/caches
            ~/.gradle/wrapper
          key: ${{runner.os}}-gradle-${{ hashFiles('**/*.gradle*', '**/gradle-wrapper.properties')}}
          restore-keys: |
            ${{ runner.os }}-gradle-
      - name: Build with Gradle
        run: ./gradlew build
```

```
module-tests:
  name: Run Module Tests
  runs-on: ubuntu-latest
  needs: build
  steps:
    - uses: actions/checkout@v4
    - name: Set up JDK 17
      uses: actions/setup-java@v4
      with:
        java-version: '17'
        distribution: 'temurin'
    - name: Cache Gradle dependencies
      uses: actions/cache@v3
      with:
        path: |
          ~/.gradle/caches
          ~/.gradle/wrapper
        key: ${{runner.os}}-gradle-${{ hashFiles('**/*.gradle*', '**/gradle-wrapper.properties')}}
        restore-keys: |
          ${{ runner.os }}-gradle-
    - name: Run Module Tests
      run: ./gradlew test
```

3.2 Результат



4 Вывод

В рамке этой лабораторной работы я написал модульные тесты по нескольким доменам. Также я упражнял написание .yml файлов для настройки `GitHub CI`.