САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Кафедра вычислительной техники

Отчёт по лабораторной работе № 1

по дисциплине «Тестирование программного обеспечения»

Вариант №2

Студенты: Куклина М.
Кириллова А.

Преподаватель:

Санкт-Петербург
2017 г.

Тестирование функции sec()

Тест

```
1 import static junit.framework.Assert.*;
2 import org.junit.Test;
3
4
5 public class testCalc {
6
7     private boolean EPS_Equals(double a, double b, double EPS)
8     {
9         return Math.abs(b - a) < EPS;
10    }
11
12    @Test
13    public void testCalcSec() {
14
15        Calc tester = new Calc();
16        try {
17
18            final double EPS = 1.0e-5, INF = 1.0e8;
19            assertTrue("at test 1:", EPS_Equals(1.0, tester.calcSec(0.0), EPS));
20            assertTrue("at test 2:", EPS_Equals(-1.0, tester.calcSec(3.0), EPS));
21            assertTrue("at test 3:", EPS_Equals(INF, tester.calcSec(Math.PI / 2), EPS));
22            assertTrue("at test 4:", EPS_Equals(INF, tester.calcSec(Math.PI / 2 * 3), EPS));
23            assertTrue("at test 5:", EPS_Equals(1/Math.cos(6000000.0), tester.calcSec(6000000.0),
      EPS));
24        }
25        catch (Exception e){
26        }
27    }
28 }
```

Тестирование модуля Fibonacci Heap

Тест

```
1 import org.junit.jupiter.api.Test;
2 import org.junit.jupiter.api.DisplayName;
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
5 import datastructures.FibonacciHeap;
6 import datastructures.FibonacciHeap.Node;
7
8 @DisplayName("Testing FibonacciHeap module.")
9 class TestFiconacciHeap {
10
11    @Test
12    @DisplayName("Testing HEAP_INSERT, EXTRACT_MIN on Integers.")
13    void testInsert() {
14        FibonacciHeap<Integer> t = new FibonacciHeap<Integer>();
15
16        for(int i = 0; i < 10; i++) {
17            t.insert((i<<i)%10);
18        }
19
20        for(Integer i = 0; i < 10; i++) {
21            assertEquals(t.minimum().key, i);
22            assertEquals(t.extractMin().key, i);
23        }
24
25    }
26    @Test
27    void testMerge() {
28        FibonacciHeap<Integer> t = new FibonacciHeap<Integer>();
29        FibonacciHeap<Integer> t1 = new FibonacciHeap<Integer>();
30
31        FibonacciHeap.Node<Integer> tMin;
32        FibonacciHeap.Node<Integer> t1Min;
33        FibonacciHeap.Node<Integer> min;
34
35        for(int i = 0; i < 10; i++) {
36            t.insert((i<<i)%10);
37        }
38
```

```
39          for(int i = 0; i < 10; i++) {
40              t1.insert((i>>i)%10);
41          }
42
43          tMin = t.minimum();
44          t1Min = t1.minimum();
45          min = tMin.compareTo(t1Min) > 0 ? t1Min : tMin;
46
47          t.merge(t1);
48
49          assertEquals(min, t.minimum());
50      }
51      @Test
52      @DisplayName("Testing DELETE")
53      void testDelete() {
54          FibonacciHeap<Integer> t = new FibonacciHeap<Integer>();
55          FibonacciHeap.Node<Integer> node5 = t.insert(5);
56          FibonacciHeap.Node<Integer> node1 = t.insert(1);
57          FibonacciHeap.Node<Integer> node3 = t.insert(3);
58          FibonacciHeap.Node<Integer> node4 = t.insert(4);
59          FibonacciHeap.Node<Integer> node2 = t.insert(2);
60
61          /* { 3, 4, 2, min(1), 5 }*/
62          t.delete(node1);
63          assertEquals(t.minimum(), node2);
64          assertEquals(t.minimum().right, node5);
65
66          t.delete(node5);
67          /* { 3, 4, min(2) }*/
68          assertEquals(t.minimum().right, node3);
69
70          t.delete(node4);
71          /* { 3, min(2) }*/
72          assertEquals(t.minimum().left, node3);
73
74          t.delete(node3);
75          /* { min(2) }*/
76          assertEquals(t.minimum(), node2);
77          assertEquals(t.minimum().left, node2);
78          assertEquals(t.minimum().right, node2);
79
80      }
81
82
83      @Test
84      @DisplayName("Testing DECREASE_KEY")
85      void testDecreaseKey() {
86          FibonacciHeap<Integer> t = new FibonacciHeap<Integer>();
87          FibonacciHeap.Node<Integer> node5 = t.insert(5);
88          FibonacciHeap.Node<Integer> node3 = t.insert(3);
89          FibonacciHeap.Node<Integer> node4 = t.insert(4);
90          FibonacciHeap.Node<Integer> node2 = t.insert(2);
91
92          t.decreaseKey(node5, 1);
93          assertEquals(t.minimum(), node5);
94
95          t.decreaseKey(node4, 0);
96          assertEquals(t.minimum(), node4);
97
98          t.decreaseKey(node3, 0);
99          assertEquals(t.minimum(), node4);
100     }
101
102
103 }
```

Тестирование доменной модели для заданной области

Текст

В первый момент показалось, что ничего не произошло, затем что-то засветилось на краю огромного экрана. По нему ползла красная звезда величиной с тарелку, а следом за ней еще одна: бинарная звездная система. Затем в углу картинки возник большой полумесяц – красный свет, переходящий в черноту – ночная сторона планеты.

UML диаграмма

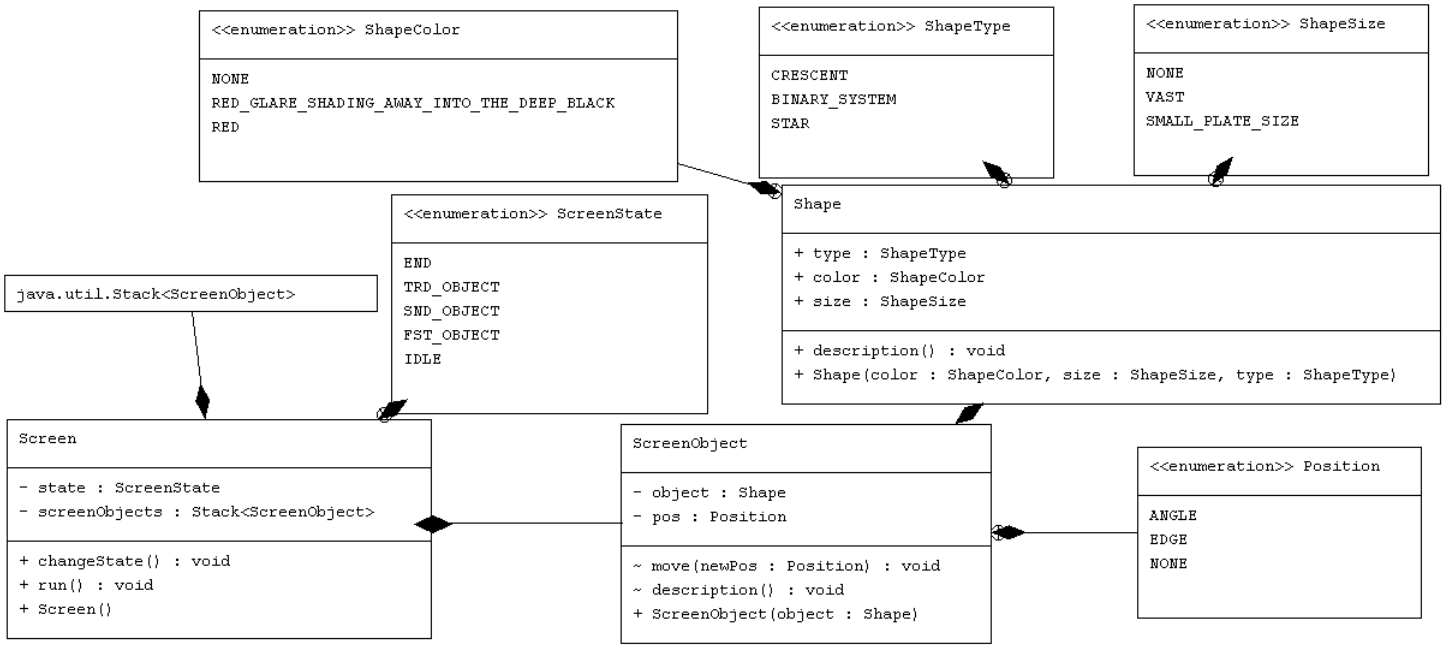

Рис. 1. UML диаграмма доменной модели

Вывод

В ходе выполнения лабораторных работ было проведено тестирование разработанных программных модулей. При выполнении работы использовались библиотеки JUnit4 и JUnit5. Явных отличий этих библиотек в данной работе отмеченно не было, разве что JUnit5 имеет иную иерахию классов и модульность, что делает его более гибким в сравнении с JUnit4, в котором все модули включены в платформу.