Национальный исследовательский университет информационных технологий, механики и оптики

Факультет ПИиКТ

Тестирование программного обеспечения

Лабораторная работа №1

Вариант №3204

Студент

Митрофанова Ирина Викторовна

Группа P33301

Преподаватель

Гаврилов Антон Валерьевич

г. Санкт-Петербург

2023

Задание

1. Для указанной функции провести модульное тестирование разложения функции в степенной ряд. Выбрать достаточное тестовое покрытие.
    - Функция arccos(x)
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.
    - Программный модуль для работы с хеш-таблицей с разрешением коллизий методом цепочек (Hash Integer, http://www.cs.usfca.edu/~galles/visualization/BucketSort.html)
3. Сформировать доменную модель для заданного текста. Разработать тестовое покрытие для данной доменной модели
    - Артур, нервничая, вошел следом и был ошеломлен, увидев развалившегося в кресле человека, положившего ноги на пульт управления и ковыряющего левой рукой в зубах правой головы. Правая голова, казалось, была всецело занята этим, но зато левая улыбалась широко и непринужденно. Количество вещей, видя которые, Артур не верил своим глазам, все росло. Его челюсть отвисла.

Выполнение:

Часть 1. Функция arccos(x)

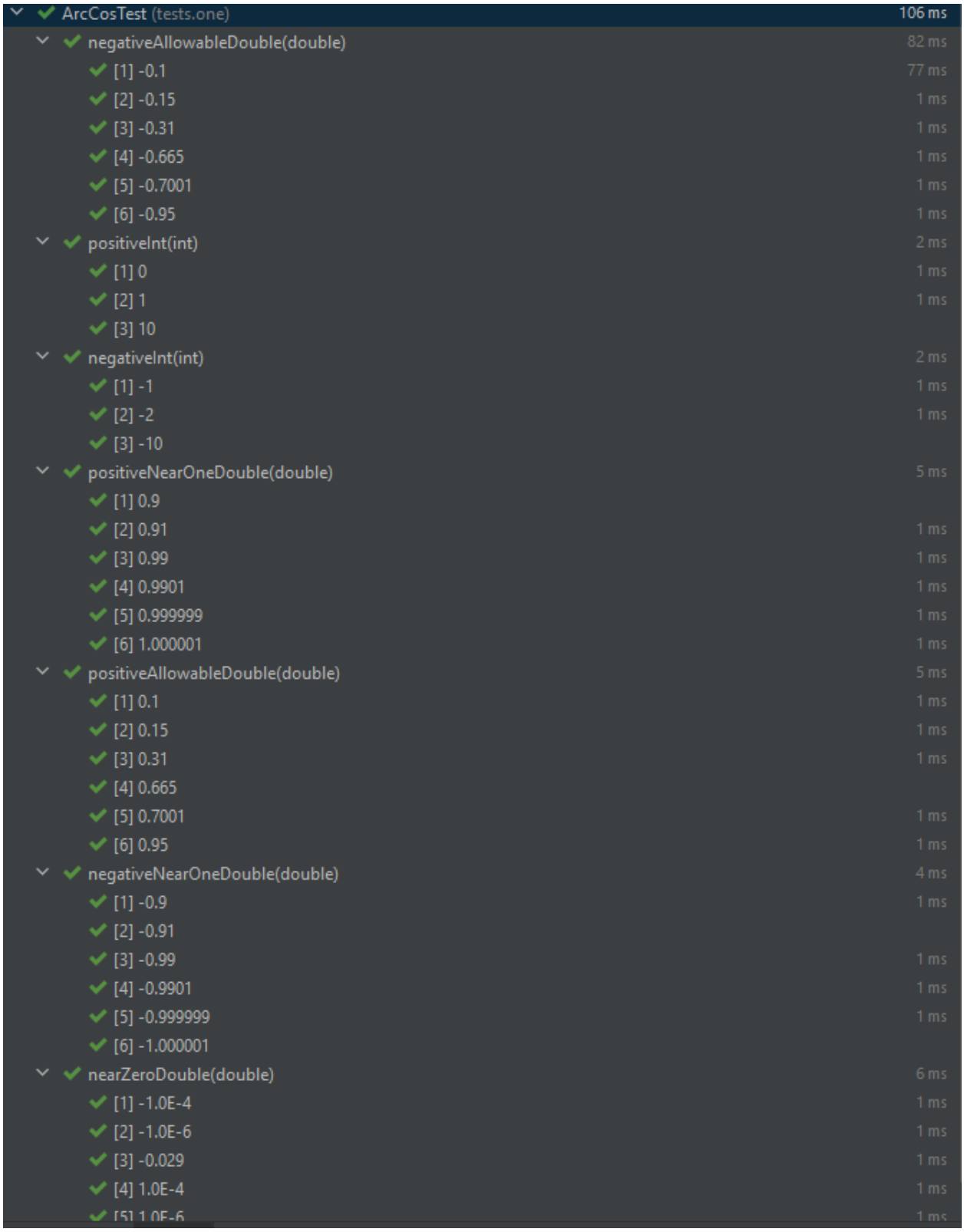
$$arccos(x) = \frac{\pi}{2} - \left[\sum_{n=1}^{\infty} \frac{1 * 3 * \ldots * (2n-1)x^{2n+1}}{2 * 4 * \ldots * 2n(2n+1)}\right]$$

```java
public static double arccos(double x) {
    if (x > 1 || x < -1) {
        return Double.NaN;
    }

    int n = 1;
    double tmp = pow(x, n * 2 + 1) / (2 * n * (2 * n + 1));
    double res = tmp;
    n++;
    while (abs(tmp) > 0.000001 && n<10000) {
        tmp = tmp * pow(x, 2) * pow((2 * n - 1), 2) / ((2 * n) * (2 * n + 1));
        res += tmp;
        n++;
    }
    res += x;
    return PI / 2 - res;
}
```

В качестве эталонных значений при тестировании использовалась функция Math.acos(x).
При сравнении результатов допускалась погрешность: *delta = 0.01*

Результаты тестирования



Часть 2. Алгоритм формирования хеш-таблицы

Реализована функция сортировки массива путём составления хеш-таблицы. Поиск индекса элемента в хеш-таблице осуществляется по формуле:

$$index(n) = \frac{n * Max\_count}{Max\_value + 1}$$

Max_count – количество элементов массива
Max_value – максимальный элемент массива

```
private int index(double val) {
    return (int)Math.floor(Math.abs(val)*size/(MAX_VALUE+1));
}
```

Структура самой хеш-таблицы представлена в виде вектора из коллекций SortedMap, где Double key – значение элемента, Integer value – количество его повторений

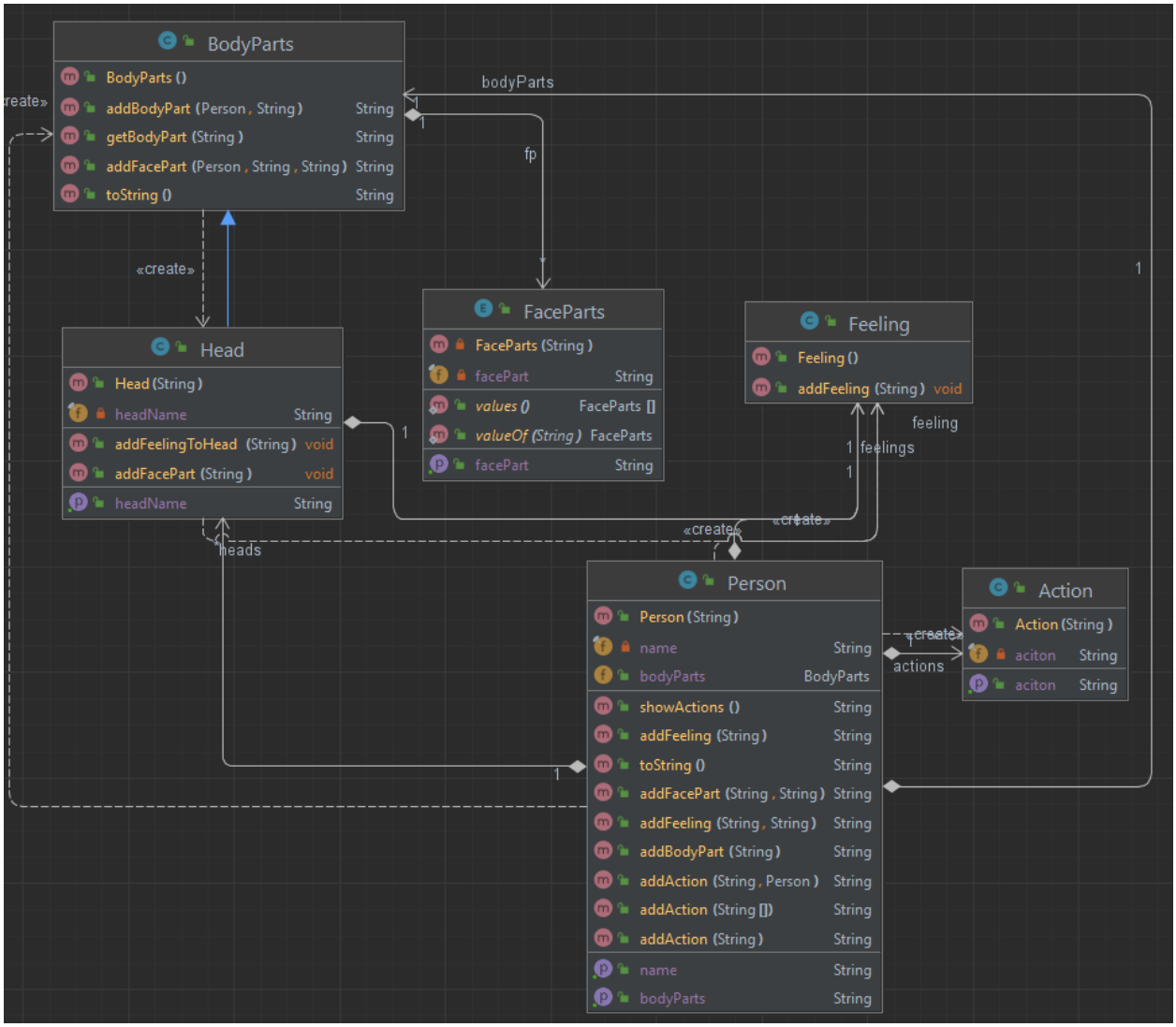
```
private Vector<SortedMap<Double, Integer>> collection;
```

В результате функция возвращает отсортированный массив

Результаты тестирования

Часть 3. Доменная область

Диаграмма классов



Оригинальный текст	Текст программы
Артур, нервничая, вошел следом и был ошеломлен, увидев развалившегося в кресле человека, положившего ноги на пульт управления и ковыряющего левой рукой в зубах правой головы. Правая голова, казалось, была всецело занята этим, но зато левая улыбалась широко и непринужденно. Количество вещей, видя которые, Артур не верил своим глазам, все росло. Его челюсть отвисла.	Артур вошёл Артур увидел Странный человек Артур не верил своим глаза Его челюсть отвисла [Артур нервничал, Артур был ошеломлён] Странный человек положил на стол ноги Странный человек ковырял левая рука в зубы правая голова правая голова была занята делом левая голова широко улыбалась

Результаты тестирования:

Вывод:

В ходе выполнения данной лабораторной работы я познакомилась с библиотекой Junit и реализовала модульное тестирование для различных алгоритмов. Для тестирования были использованы обычные (@Test) и параметризированные (@ParameterizedTest) тесты.

Исходный код:
https://github.com/LFiosx18/TPO/tree/main/Lab1