Университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04

Программная инженерия

Дисциплина «Тестирование программной инженерии»

Отчет

по лабораторной работе № 1

Выполнил:

Савельева Диана Александровна

P33082

Преподаватель:

[Харитонова](https://my.itmo.ru/persons/102300) Анастасия Евгеньевна

Санкт-Петербург, 2024 г

***Задание***

* Для указанной функции провести модульное тестирование разложения функции в степенной ряд. Выбрать достаточное тестовое покрытие.

*Функция cos(x)*

* 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.

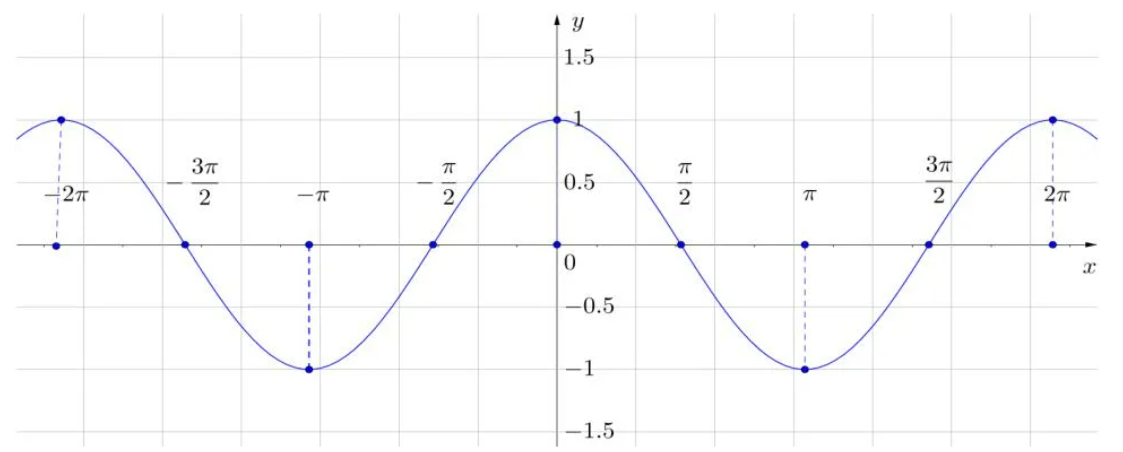
*Программный модуль для работы c хеш-таблицей с закрытой адресацией (Hash String,*[*http://www.cs.usfca.edu/~galles/visualization/OpenHash.html*](http://www.cs.usfca.edu/~galles/visualization/OpenHash.html)*)*

* Сформировать доменную модель для заданного текста.  Разработать тестовое покрытие для данной доменной модели.

*И поскольку это далеко не самое естественное положение для кита, то у этого несчастного существа было очень мало времени на то, чтобы успеть свыкнуться с осознанием того, что оно кит, перед тем, как ему пришлось свыкнуться с осознанием того, что оно уже больше не кит.*

***Задание №1.***

* Область определения функции y = cos (x) не имеет ограничений для x.
* Область значений y = cos(x) D(y): [-1;1].



Тестовое покрытие выбрано в следующих точках:

doubles = {-2 \* Math.PI, -1.5 \* Math.PI, -Math.PI, -0.5 \* Math.PI, 0, 0.5 \* Math.PI, Math.PI, 1.5 \* Math.PI, 2 \* Math.PI}

**Здесь рассмотрены все основные точки для cos.**

doubles = {Double.POSITIVE\_INFINITY, Double.NEGATIVE\_INFINITY, Double.NaN}

**Здесь рассмотрены все некорректные значения для cos.**

*Код модуля:*

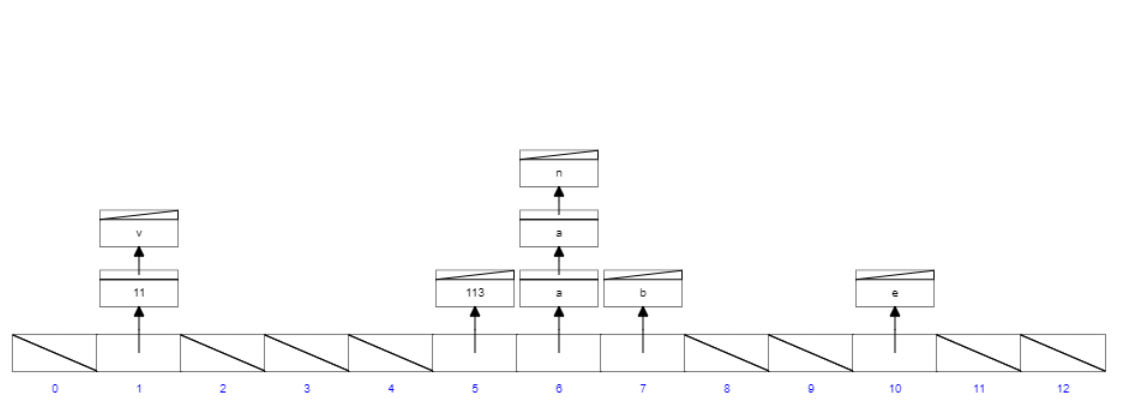
<https://github.com/ITSamantha/software-testing/tree/main/lab1/src/main/task1>

*Код тестирования:*

https://github.com/ITSamantha/software-testing/tree/main/lab1/src/test/task1

***Задание №2.***

* Реализована HashTable с закрытой адресацией (наличие chain у индекса).
* Реализован собственный LinkedList для chain.
* Реализован Node для хранения узла и значения для LinkedList.



Проводилось тестирование для LinkedList и HashTable.

Для LinkedList были протестированы методы delete, insert, contains и др.

Для HashTable были протестированы методы delete, insert и др.

*Код модуля:*

https://github.com/ITSamantha/software-testing/tree/main/lab1/src/main/task2

*Код тестирования:*

https://github.com/ITSamantha/software-testing/tree/main/lab1/src/test/task2

***Задание №3.***

* Реализованы следующие enum:
  + CreatureType
  + PositionType
  + StateType
  + TimeType
* Реализованы следующие интерфейсы:
  + ICatchable
* Реализованы следующие классы:
  + Awareness
  + Creature
  + Main
  + Whale

Протестированы классы Whale, Awareness. Для тестирования использовались Enum.

*Код модуля:*

https://github.com/ITSamantha/software-testing/tree/main/lab1/src/main/task3

*Код тестирования:*

https://github.com/ITSamantha/software-testing/tree/main/lab1/src/test/task3

***Вывод:***

В ходе выполнения работы я познакомилась с инструментом тестирования ПО JUNIT для Java. Попробовала создавать как полностью ручные тесты Test, так и ParameterizedTest. Во время работы неоднократно тесты помогли выявить ошибки в работе кода.