Университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04

Программная инженерия

Дисциплина «Тестирование программной инженерии»

Отчет

по лабораторной работе № 2

Выполнил:

Савельева Диана Александровна

P33082

Преподаватель:

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

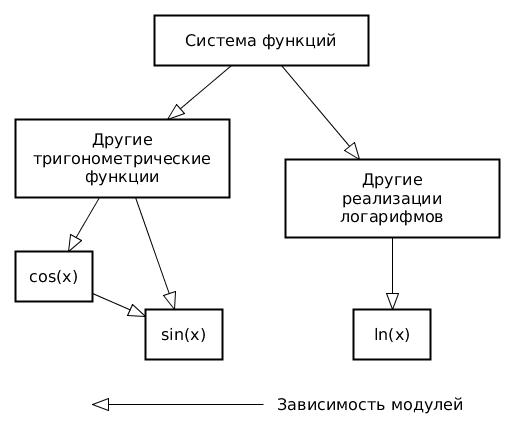
Санкт-Петербург, 2024 г

Вариант № 117359

1. **Задание:**

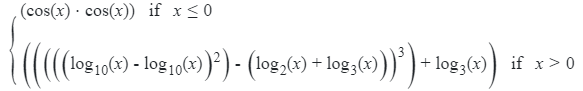
*Правила выполнения работы:*

1. Все составляющие систему функции (как тригонометрические, так и логарифмические) должны быть выражены через базовые (тригонометрическая зависит от варианта; логарифмическая - натуральный логарифм).
2. Структура приложения, тестируемого в рамках лабораторной работы, должна выглядеть следующим образом (пример приведён для базовой тригонометрической функции sin(x)):



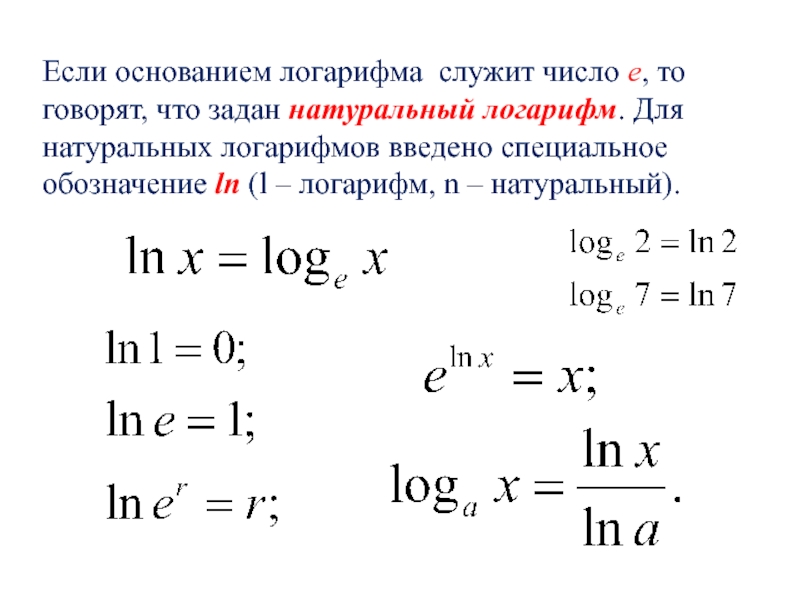
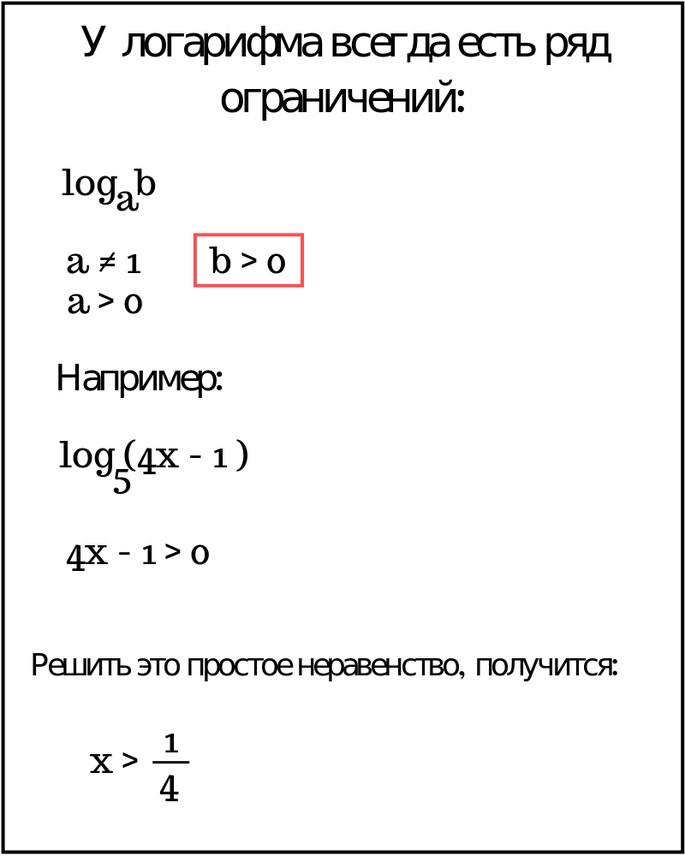
1. Обе "базовые" функции (в примере выше - sin(x) и ln(x)) должны быть реализованы при помощи разложения в ряд с задаваемой погрешностью. Использовать тригонометрические / логарифмические преобразования для упрощения функций ЗАПРЕЩЕНО.
2. Для КАЖДОГО модуля должны быть реализованы табличные заглушки. При этом, необходимо найти область допустимых значений функций, и, при необходимости, определить взаимозависимые точки в модулях.
3. Разработанное приложение должно позволять выводить значения, выдаваемое любым модулем системы, в сsv файл вида «X, Результаты модуля (X)», позволяющее произвольно менять шаг наращивания Х. Разделитель в файле csv можно использовать произвольный.

*Система функций:*

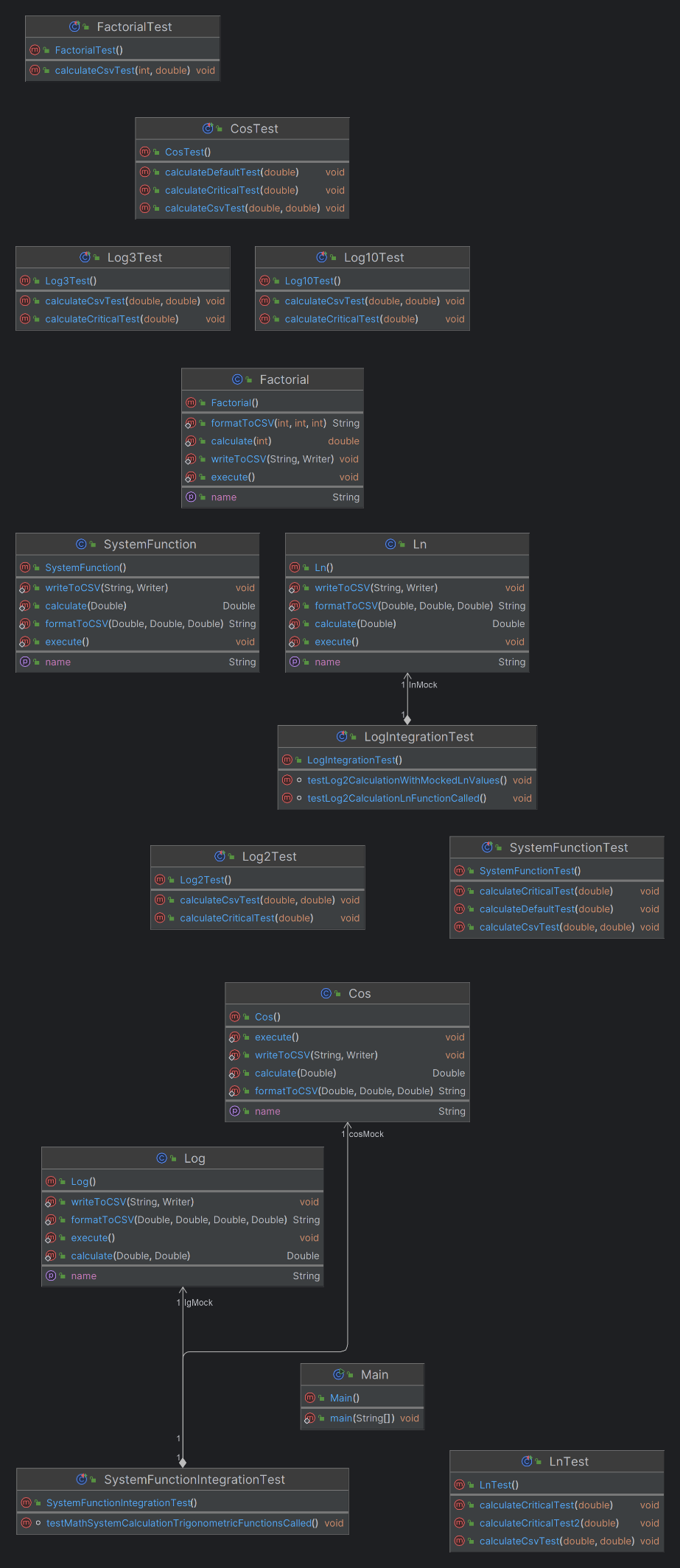


Некоторые замечания:

* Для нахождения log a (b) используем следующие свойства логарифма.



**2. Диаграмма классов**

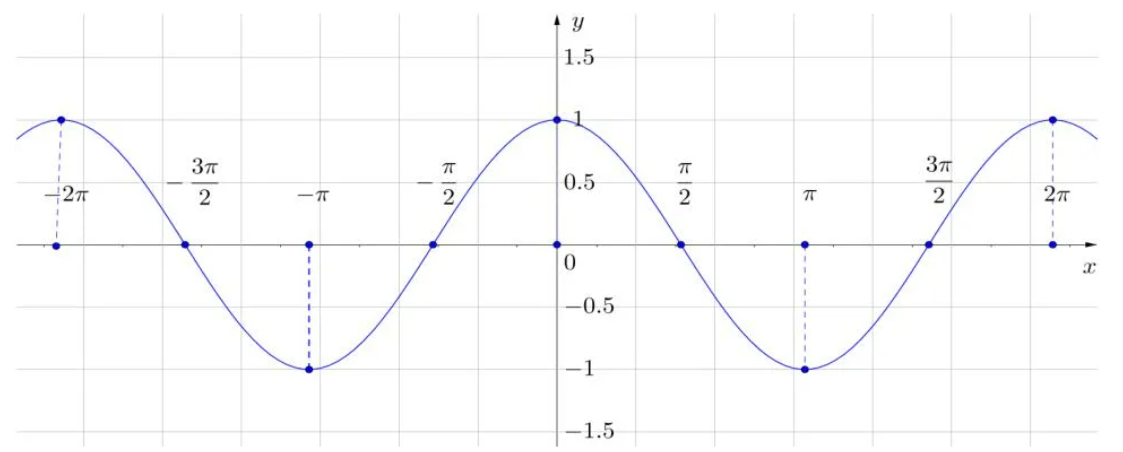
****

**3. Тестовое покрытие**

Тестовое покрытие для каждого из модулей можно просмотреть в репозитории на GitHub. Тестовое покрытие выбрано таким образом, чтобы покрыть все возможные вводимые пользователем значения, а также критические точки.

Например, для cos.

* 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:*

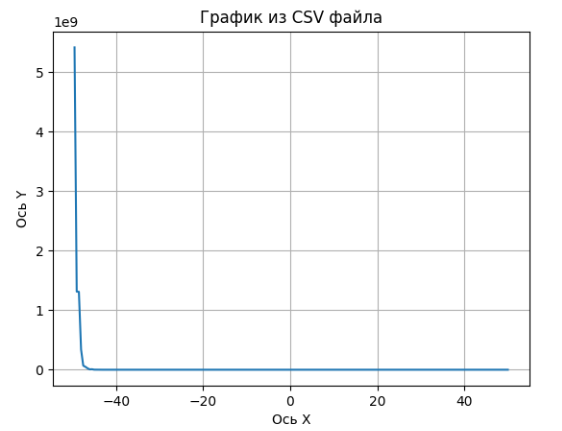
<https://github.com/ITSamantha/software-testing/tree/main/lab1/src/main/task1>

*Код тестирования:*

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

Для всех модулей сформированы csv файлы со значениями x и y, которые и используются для тестов. Для некоторых модулей намерено сделаны ошибки для проверки тестов.

График результирующей функции представлен на рисунке:



***Вывод:***

В ходе выполнения работы я познакомилась с инструментом тестирования ПО JUNIT для Java, а также использовала модуль Mockito. Попробовала создавать как полностью ручные тесты Test, так и ParameterizedTest. Во время работы неоднократно тесты помогли выявить ошибки в работе кода. Также попробовала интеграционное тестирование.