

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«Санкт-Петербургский национальный исследовательский университет  
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И  
КОМПЬЮТЕРНОЙ ТЕХНИКИ

**ЛАБОРАТОРНАЯ РАБОТА №2**  
**по дисциплине**  
**«Тестирование программного обеспечения»**

Вариант №3412345

***Выполнил:***  
Студент группы Р3334  
Баянов Равиль  
Динарович  
***Преподаватель:***  
Бострикова Дарья  
Константиновна

Санкт-Петербург  
2025

# Оглавление

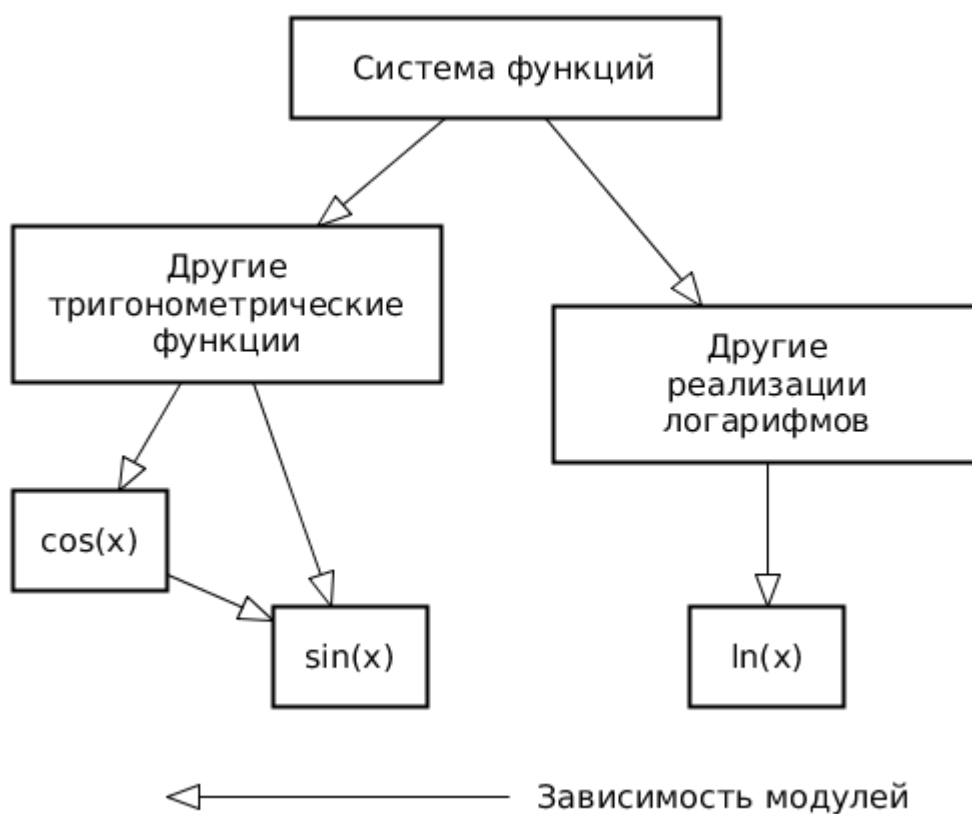
<b>Задание.....</b>	<b>3</b>
<b>UML-диаграмма .....</b>	<b>5</b>
<b>Описание тестового покрытия.....</b>	<b>6</b>
<b>Графики на основе csv-выгрузок.....</b>	<b>8</b>
<b>Вывод .....</b>	<b>10</b>

# Задание

Провести интеграционное тестирование программы, осуществляющей вычисление системы функций (в соответствии с вариантом).

## 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  $\sin(x)$ ):



3. 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.
4. Для КАЖДОГО модуля должны быть реализованы табличные заглушки. При этом, необходимо найти область допустимых значений функций, и, при необходимости, определить взаимозависимые точки в модулях.

5. Разработанное приложение должно позволять выводить значения, выдаваемое любым модулем системы, в csv файл вида «X, Результаты модуля (X)», позволяющее произвольно менять шаг наращивания X. Разделитель в файле csv можно использовать произвольный.

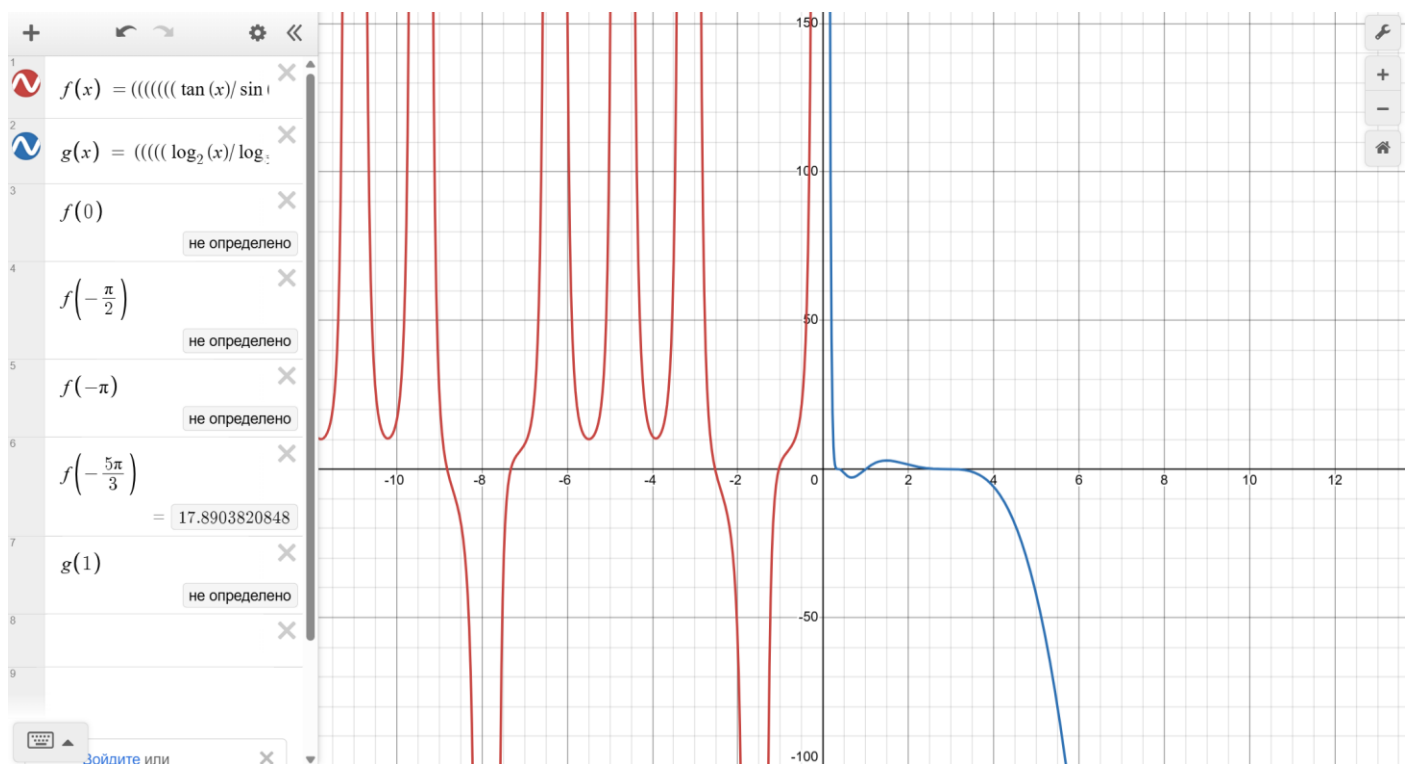
## Вариант:

$$\begin{cases} \left( \left( \left( \left( \left( \frac{\tan(x)}{\sin(x)} \right)^2 - \sec(x) + (\tan(x) + (\cot(x) \cdot (\cot(x) \cdot \csc(x)))) \right) \right) \right) \right) \frac{\tan(x) \cdot (\cos(x)^2)}{\sec(x)} \right) - ((\sec(x) - \cos(x))^3) + \left( \sec(x) + \left( \left( \frac{\cot(x)}{\sec(x)} \right) \cdot (\sin(x) - (\cos(x) - \cos(x))) \right) \right) \right) & \text{if } x \leq 0 \\ \left( \left( \left( \left( \frac{\log_2(x)}{\log_5(x)} \right) - (\ln(x) - \ln(x)) \right) - (\log_2(x)^2)^3 \right) \cdot \log_3(x) \right) & \text{if } x > 0 \end{cases}$$

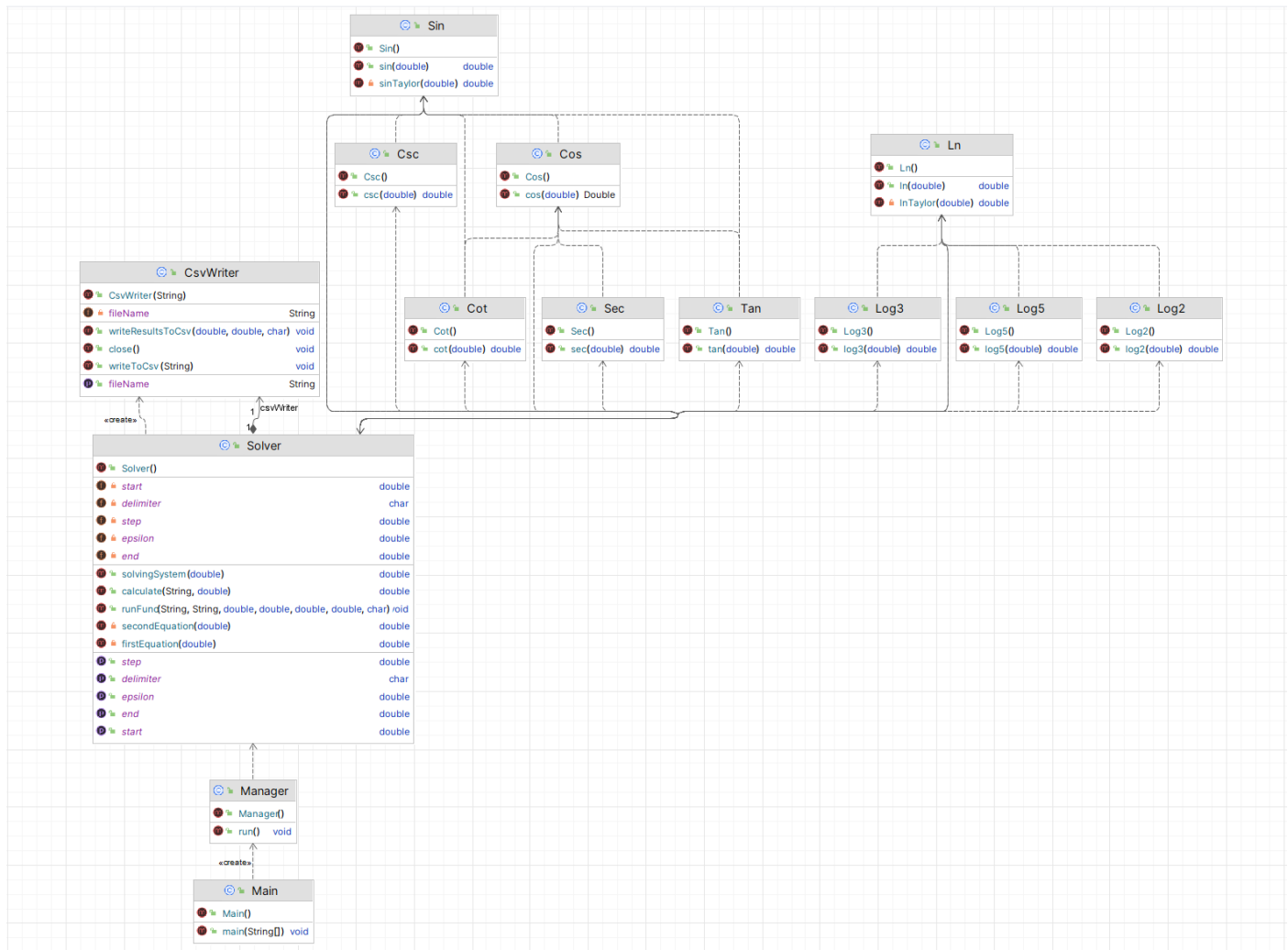
$x \leq 0$  : (((((((tan(x) / sin(x)) ^ 2) - sec(x)) + (tan(x) + (cot(x) \* (cot(x) \* csc(x)))))) / ((tan(x) \* (cos(x) ^ 2)) / sec(x))) - ((sec(x) - cos(x) ^ 3)) + (sec(x) + ((cot(x) / sec(x)) \* (sin(x) - (cos(x) - cos(x)))))

$x > 0$  : ((((((log\_2(x) / log\_5(x)) - (ln(x) - ln(x))) - (log\_2(x) ^ 2) ^ 3) \* log\_3(x))

## График функций из варианта:



# UML-диаграмма



# Описание тестового покрытия

**ОДЗ:** заметим, что в моей функции  $\sin(x)$  и  $\cos(x)$  не должны быть равны 0. Следовательно, ОДЗ для тригонометрической функции:

$x \neq -\frac{\pi}{2}k$ , где  $k$  – целое число.

Для тригонометрической функции из-за логарифма  $x$  не должен быть меньше или равен нулю. Но также мы исключаем 1, так как происходит деление на логарифм.  
 $x \in (0, 1) \cup (1, +\infty)$

Будем пользоваться стратегией интеграции сверху-вниз.

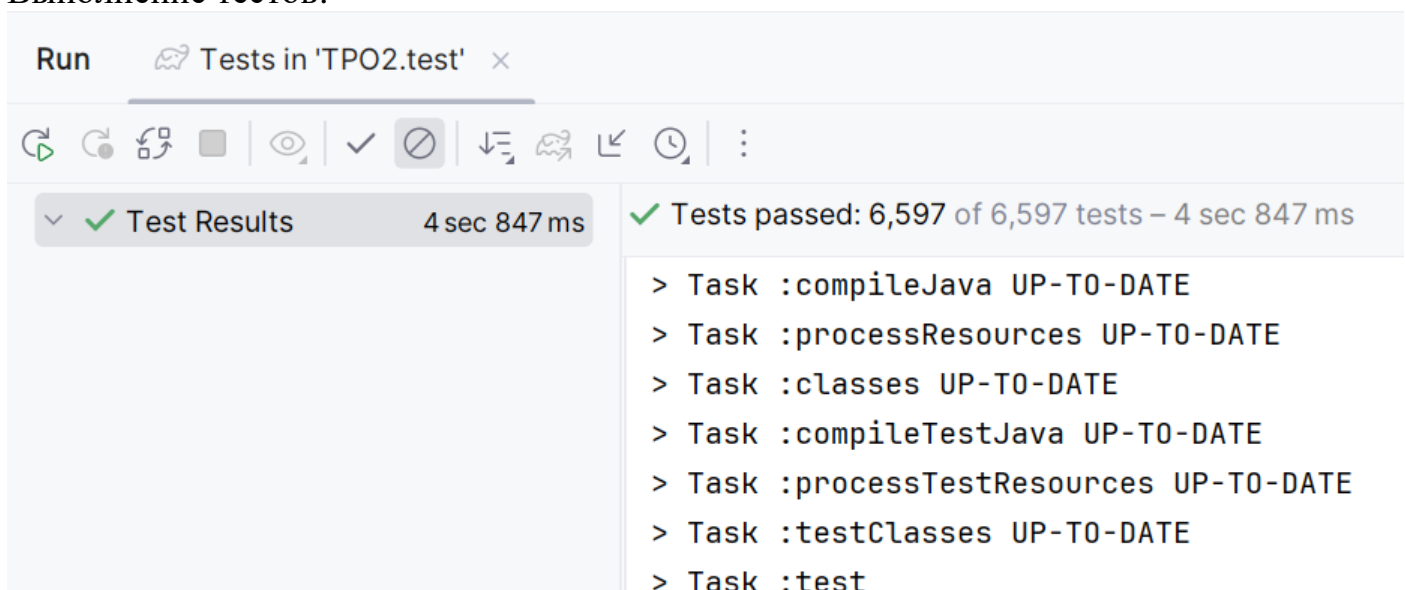
Также напишем модульные тесты для каждой функции и также протестируем функции на периодичность.

На нашем графике есть также особые точки, которые мы сейчас рассмотрим. Так как наша тригонометрическая функция периодична со значением  $2\pi$ . Рассмотрим точки в диапазоне  $[-2\pi; 0]$ . В диапазоне  $[-\frac{1}{3}\pi; 0)$  функция возрастает. В точке 0 функция принимает значение  $+\infty$ .  $(-\frac{1}{2}\pi; -\frac{1}{3}\pi]$  – функция также возрастает. В точке  $-\frac{1}{2}\pi$  – функция равна  $-\infty$ . В диапазоне  $[-\frac{4}{5}\pi; -\frac{1}{2}\pi)$  – функция убывает. В диапазоне  $(-\pi; -\frac{4}{5}\pi]$  – функция убывает и в точке  $-\pi$  – функция равна  $+\infty$ .  $[-1.25\pi; -\pi)$  – возрастание.  $(-\frac{3}{2}\pi; -1.25\pi]$  – убывание до точки перегиба.  $[-\frac{7}{4}\pi; -\frac{3}{2}\pi)$  – возрастание.  $(-2\pi; -\frac{7}{4}\pi]$  – убывание.

Для логарифмической функции имеем две точки перегиба 0.67085 и 1.49065. До 0.67085 функция убывает, до 1.49065 – возрастает, а после убывает до бесконечности. Также исключаем точку 1, так как она не входит в ОДЗ.

На основе этих диапазонов и этих особых точек составим классы эквивалентности и напишем тесты для табличных значений, для этих особых точек и для этих диапазонов, проверяя их на возрастание и убывание. Напишем модульные тесты, для каждой из функций и проверим их работу на этих точках. И обязательно проверим нашу общую функцию на периодичность.

Выполнение тестов:

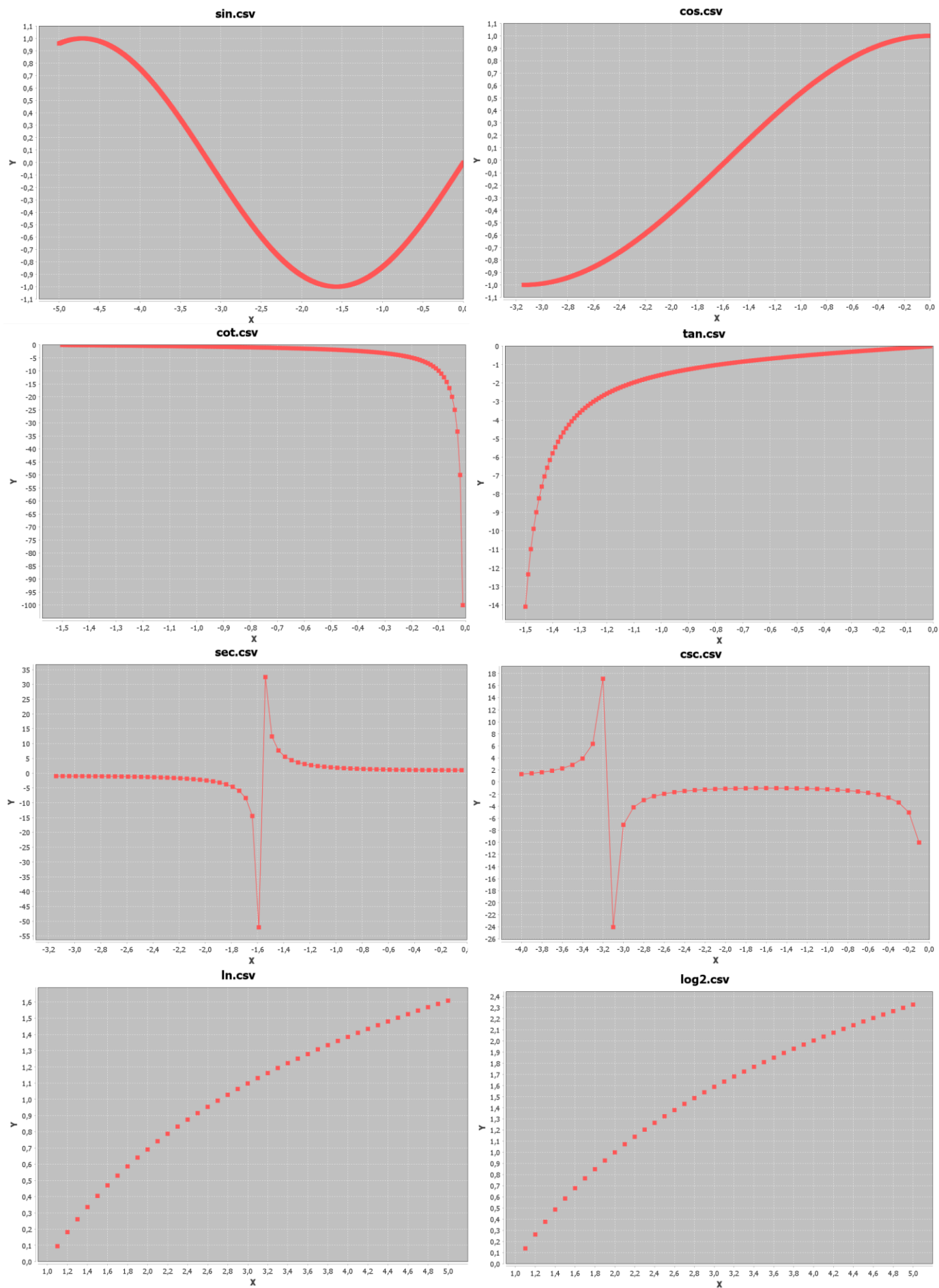


The screenshot shows an IDE window titled 'Run Tests in 'TPO2.test' x'. Below the title bar is a toolbar with icons for running, debugging, and other actions. The main area is divided into two panes. The left pane, titled 'Test Results', shows a green checkmark and the text '4 sec 847 ms'. The right pane shows a green checkmark and the text 'Tests passed: 6,597 of 6,597 tests – 4 sec 847 ms'. Below this, a list of tasks is displayed:

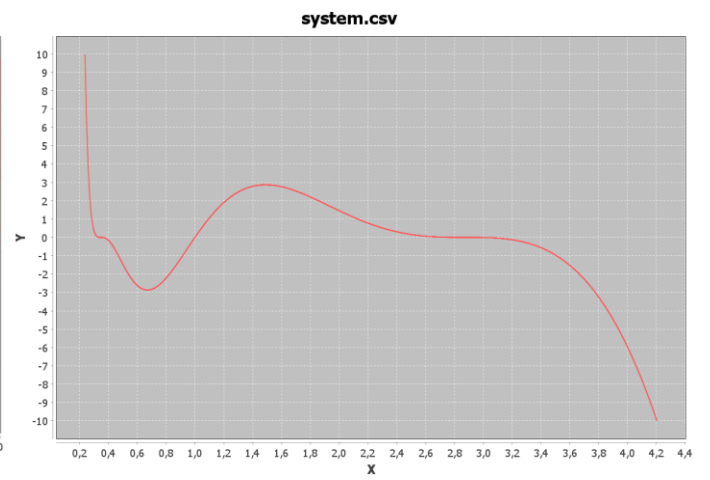
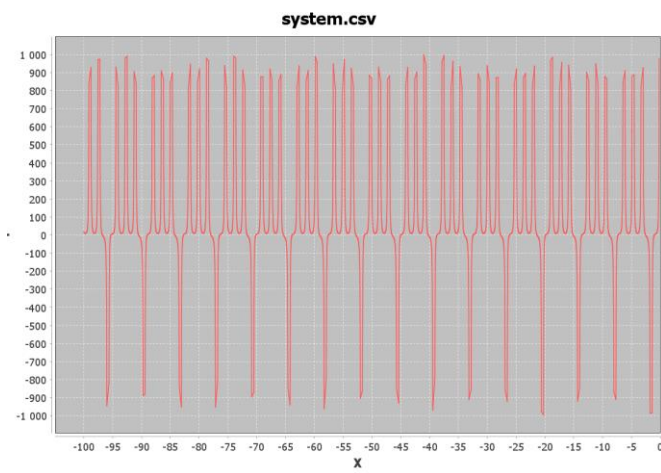
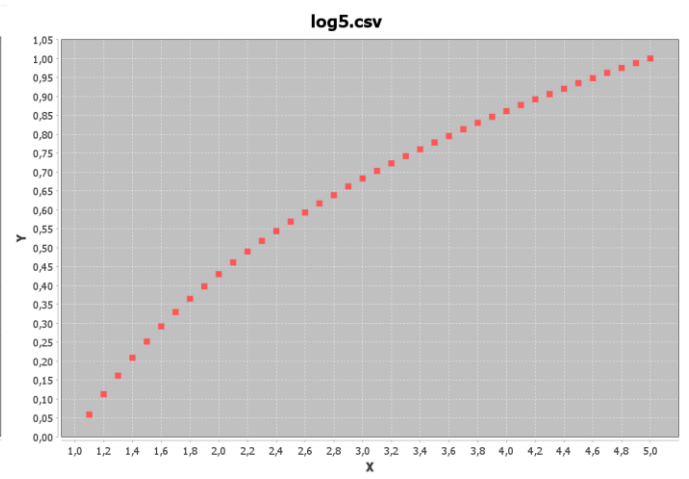
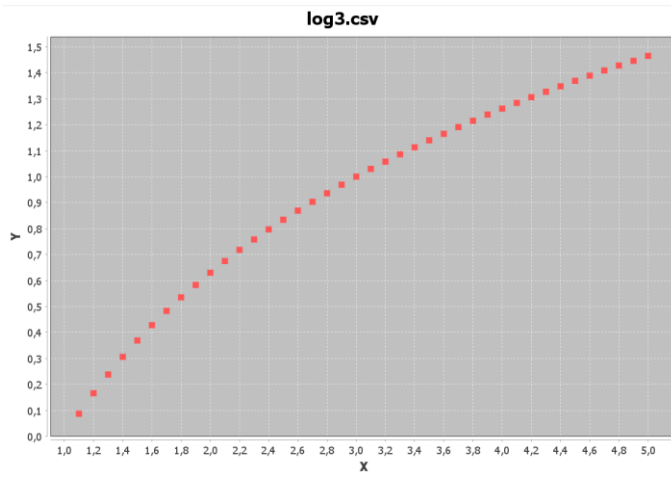
```
> Task :compileJava UP-TO-DATE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE
> Task :compileTestJava UP-TO-DATE
> Task :processTestResources UP-TO-DATE
> Task :testClasses UP-TO-DATE
> Task :test
```

Код: <https://github.com/RavvChek/TPO2/tree/master>

# Графики на основе csv-выгрузок







# Вывод

Выполнив данную лабораторную работу, я чуть ближе познакомился с интеграционным тестированием, тщательно проанализировал функцию и свою программу и протестировал её при каждом возможном случае.