Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»
Факультет программной инженерии и компьютерной техники

Вариант №77
Лабораторная работа №2
по дисциплине
ТПО

Выполнил студент группы P3312
Соколов А.В.
Пархоменко К.А.
Преподаватель:
Кривоносов Е.Д.

г. Санкт-Петербург
2025г.

Содержание

1 Задание

1.1 Задание

$$x \leq 0 : (((((\sec(x) \times \csc(x))/\cos(x))\sec(x))^2)\sin(x))$$

$$x > 0 : (((((\log_2(x) + \log_{10}(x))^2)\log_2(x))\log_{10}(x))\log_5(x))$$

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.

2 Выполнение

2.1 Анализ домена и возможных данных

Анализ домена:

$$\mathbb{R} \setminus \{\{0\} \cup \{-k\frac{\pi}{2}\}_{k=1}^{\infty}\}$$

что эквивалентно

$$(\ldots \cup (-\pi, -\frac{\pi}{2}) \cup (-\frac{\pi}{2}, 0) \cup (0, \infty))$$

получаем неопределенность в $x = 0$ и $x = -\frac{\pi}{2}, -\pi, -\frac{3\pi}{2}, \ldots$.

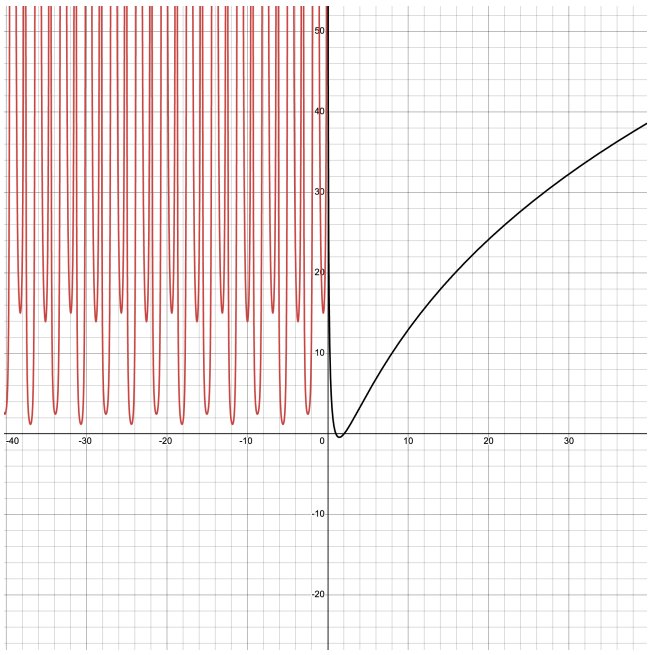

Рис. 1: Система функций

2.2 UML-диаграмму классов разработанного приложения

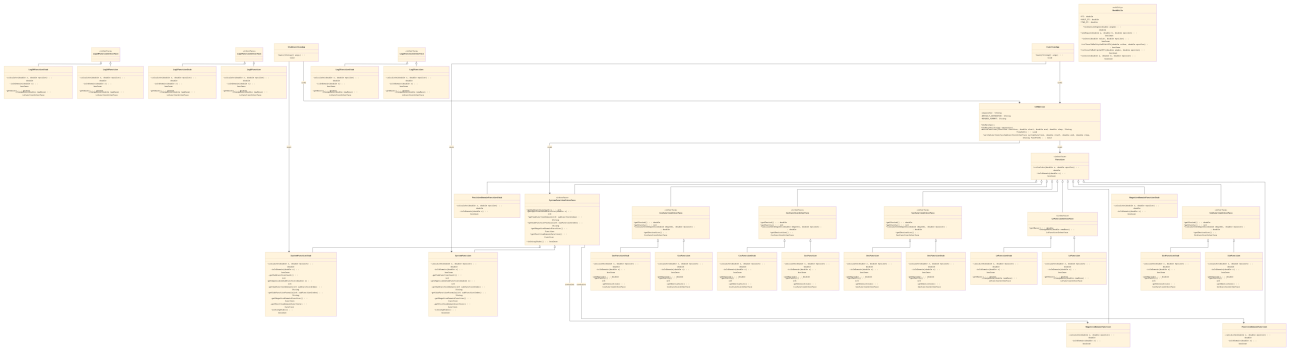

Рис. 2: FunctionApp диаграмма

2.3 Описание тестового покрытия с обоснованием его выбора

Мы решили использовать **bottom-up интеграционное тестирование** для этого проекта. Потому что у нас довольно сложная структура, где одни компоненты зависят от других. Начинаем с самых простых кирпичиков и постепенно строим из них что-то более сложное.

2.3.1 Базовые функции

Проверяем, правильно ли работают sin(x) и ln(x), используя их разложение в ряд.

2.3.2 Производные тригонометрические функции

Подключаем cos(x) к sin(x) и смотрим, как они работают вместе. Подключаем sec(x) к cos(x). Подключаем csc(x) к sin(x).

2.3.3 Производные логарифмические функции

Подключаем log2, log10, log5 к ln.

2.3.4 Функции для конкретных доменов

Подключаем NegativeDomainFunction к тригонометрическим функциям. Подключаем PositiveDomainFunction к логарифмическим функциям.

2.3.5 Интеграция системной функции

Собираем все вместе: SystemFunction, NegativeDomainFunction и PositiveDomainFunction.

2.4 Подробности тестирования функций Sin и Ln

2.4.1 Тестирование SinFunction

- **Проверка стандартных углов**
- **Подтверждение нечетности**: Sin(–x) = –Sin(x).
- **Проверка периодичности**: Sin(x) = Sin(x + 2π).
- **Ограничение значений**: Sin всегда выдает значения в диапазоне от –1 до 1.
- **Проверка непрерывности**

2.4.2 Тестирование LnFunction

- **Проверка стандартных значений**: Как и с Sin, мы проверяем Ln на известных значениях, сравнивая результат с заранее вычисленными значениями.
- **Поведение вблизи 1**: Мы тестируем поведение Ln около 1, где используется приближение через ряд Тейлора.
- **Обработка малых и больших значений**: Проверяем, как Ln ведет себя, когда на вход подаются очень маленькие или очень большие числа.
- **Проверка домена определения**
- **Проверка логарифмических тождеств**: Мы проверяем ключевые свойства логарифмов:
  1. $\ln(a \cdot b) = \ln(a) + \ln(b)$
  2. $\ln(x^n) = n \cdot \ln(x)$

2.4.3 Тестирование системной функции (SystemFunctionIntegrationTest)

1. **Проверка домена**: Убеждаемся, что функция правильно определяет домен данных (‘testIsInDomain()‘).
2. **Параметризованные тесты**: Подставляем разные числа и смотрим, выдает ли функция правильные ответы (‘testCalculate(double x, double expected)‘).
3. **Граничные значения**: Проверяем, как функция ведет себя на границах допустимых значений (‘testSystemFunctionAtBoundaries()‘).
4. **Доменное разделение**: Отдельно тестируем функцию для положительных и отрицательных чисел (‘testNegativeDomainCalculation()‘ и ‘testPositiveDomainCalculation()‘).

2.4.4 Тестирование вывода в CSV (CSVOutputIntegrationTest)

1. **Корректность файлов**: Проверяем, правильно ли формируются CSV-файлы с данными.
2. **Гибкость шага**: Тестируем с разными размерами шага между значениями в CSV-файле (‘testCSVWriterWithDifferentStepSizes()‘).
3. **Обработка ошибок**: Проверяем, как система реагирует на неправильные входные данные (‘testCSVWriterWithInvalidInputs()‘).

2.5 Результат работы программы

lab2 > out.csv > data

```
1   X,F(X)
2   -100.0,1.7321425631964833
3   -99.0,367263.44226097426
4   -98.0,14.01009271847223
5   -97.0,4.367254938051253
6   -96.0,660.8238363215764
7   -95.0,17.613634350621254
8   -94.0,10.690228253673398
9   -93.0,52.567213214981535
10  -92.0,22.893447910866538
11  -91.0,72.97954504346208
12  -90.0,12.047171873382213
13  -89.0,42.16888926908222
14  -88.0,857.7059083688779
15  -87.0,3.151529866843396
16  -86.0,98.30902082727125
17  -85.0,47.11647700061499
18  -84.0,2.920260315672122
19  -83.0,424.9589856596287
20  -82.0,21.407472333473805
21  -81.0,1.177079939246862
```

Рис. 3: CSV файл

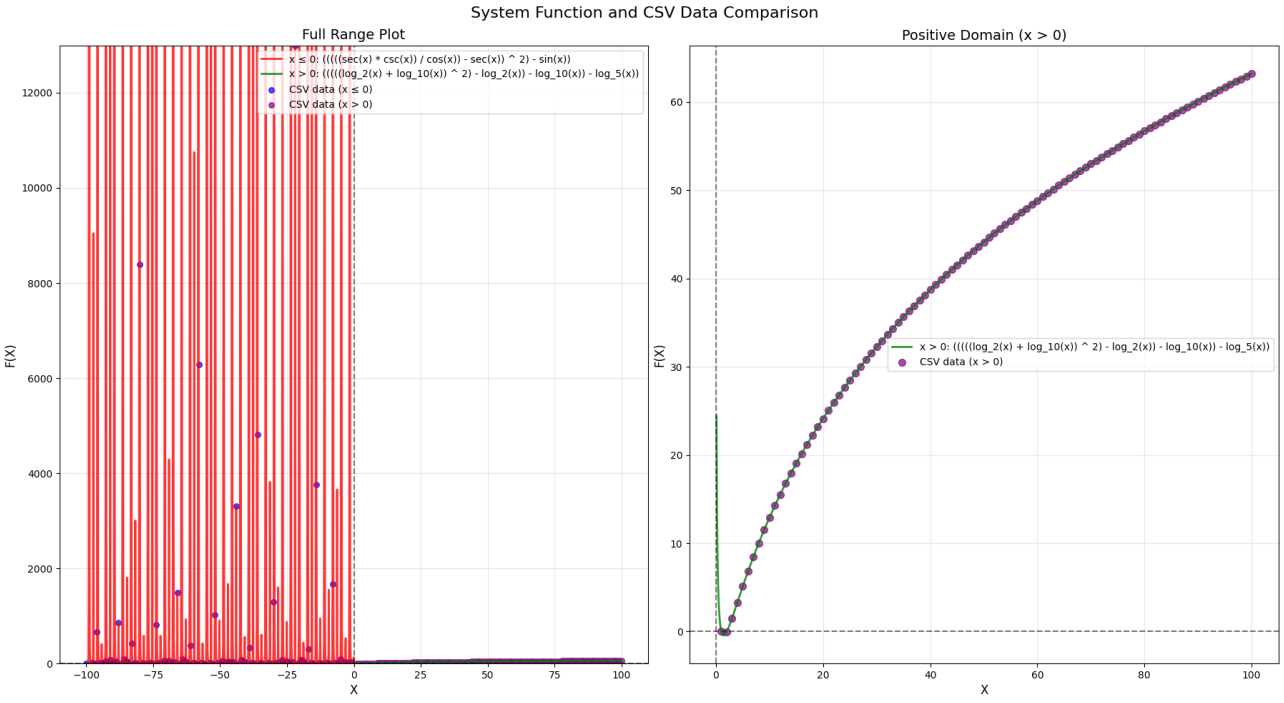

Рис. 4: Результат выполнения

2.6 Вывод

Для реализации системы функций я использовал разложение в ряд с заданной погрешностью для базовых тригонометрических и логарифмических функций.

Для каждого модуля системы я реализовал табличные заглушки. Я определил область допустимых значений функций и взаимозависимые точки между модулями.

Разработанное мной приложение обеспечивает вывод результатов работы любого модуля в CSV-файл с возможностью изменения шага наращивания аргумента X. Формат файла "X, Результаты модуля (X)"с произвольным разделителем.

Общая структура приложения соответствует примеру, приведенному в условии задания, где базовые функции $\sin(x)$ и $\ln(x)$ реализованы через разложение в ряд.