Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
Факультет программной инженерии и компьютерной техники

# Тестирование программного обеспечения
Лабораторная №2

Вариант 2918

Выполнил:
Беляков Дмитрий
Группа:
P33122
Преподаватель:
Харитонова А. Е.

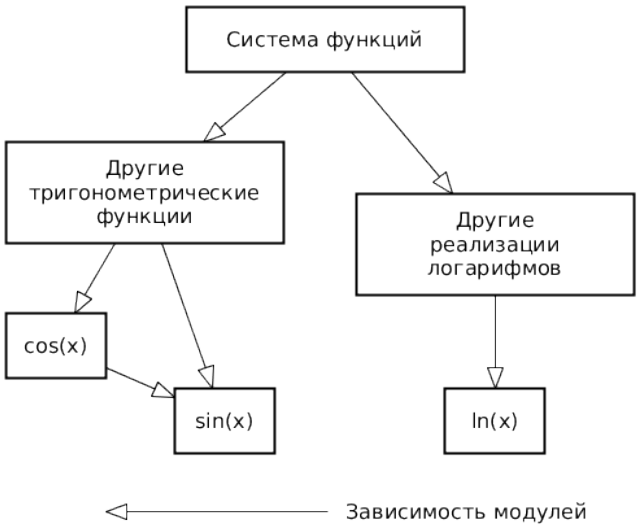
Санкт-Петербург
2021

**Задание**

$$\begin{cases} \left(\left(\left(\left(\frac{\sec(x)\cdot\cos(x)}{\sin(x)}\right)+\cos(x)\right)+\sin(x)\right)\cdot\csc(x)\right) & \text{if } x \le 0 \\ \left(\left(\left(\frac{(\log_{10}(x)+\log_3(x))\cdot\log_2(x)}{\ln(x)}\right)+\log_5(x)\right)+\log_5(x)\right) & \text{if } x > 0 \end{cases}$$

Все составляющие систему функции (как тригонометрические, так и логарифмические) должны быть выражены через базовые (тригонометрическая зависит от варианта; логарифмическая - натуральный логарифм).

1.Структура приложения, тестируемого в рамках лабораторной работы, должна выглядеть следующим образом (пример приведён для базовой тригонометрической функции sin(x)):




2.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.

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

4.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.

**Порядок выполнения работы:**

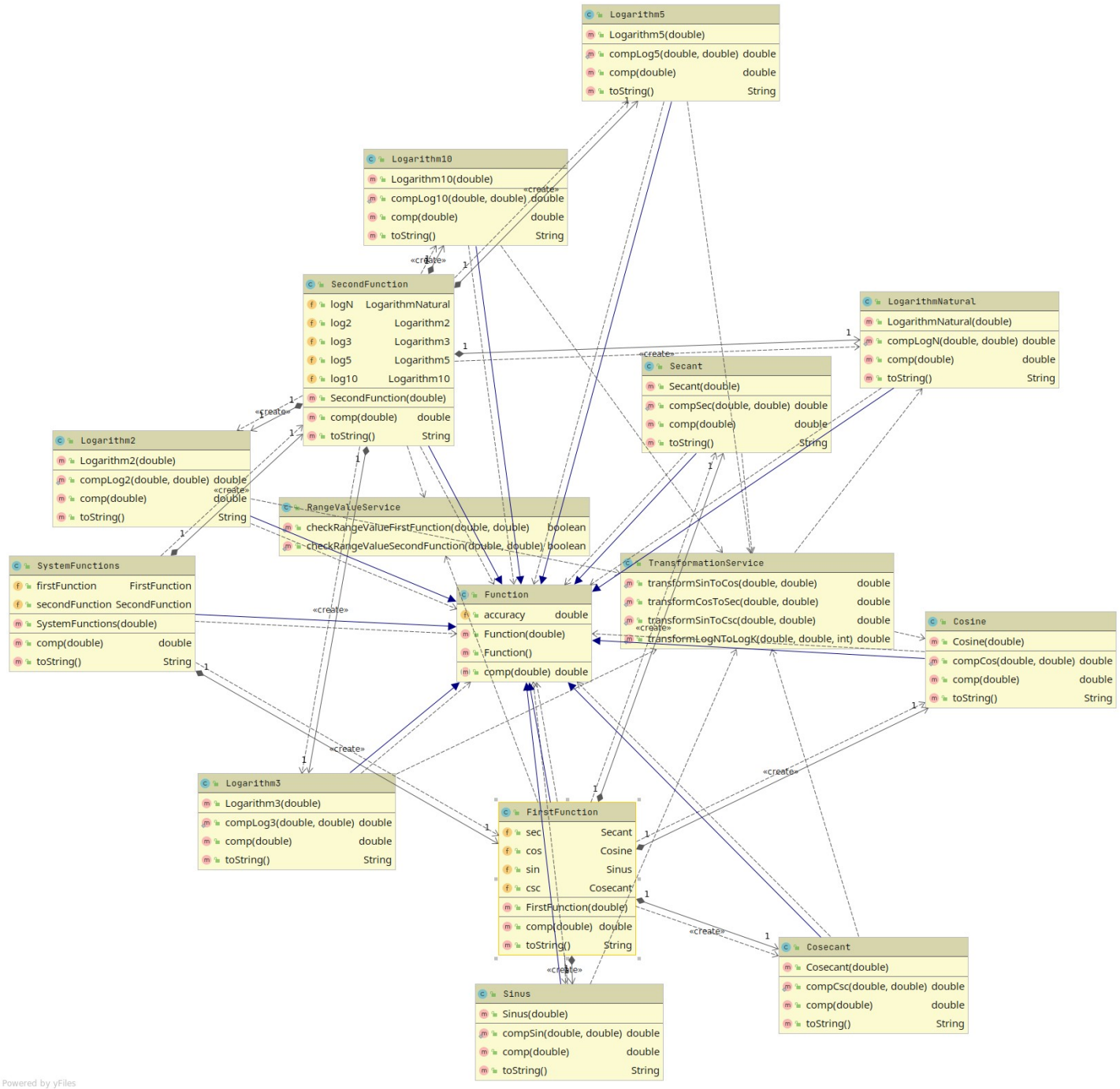
1.Разработать приложение, руководствуясь приведёнными выше правилами.

2.С помощью JUNIT4 разработать тестовое покрытие системы функций, проведя анализ эквивалентности и учитывая особенности системы функций. Для анализа особенностей

системы функций и составляющих ее частей можно использовать сайт https://www.wolframalpha.com/.

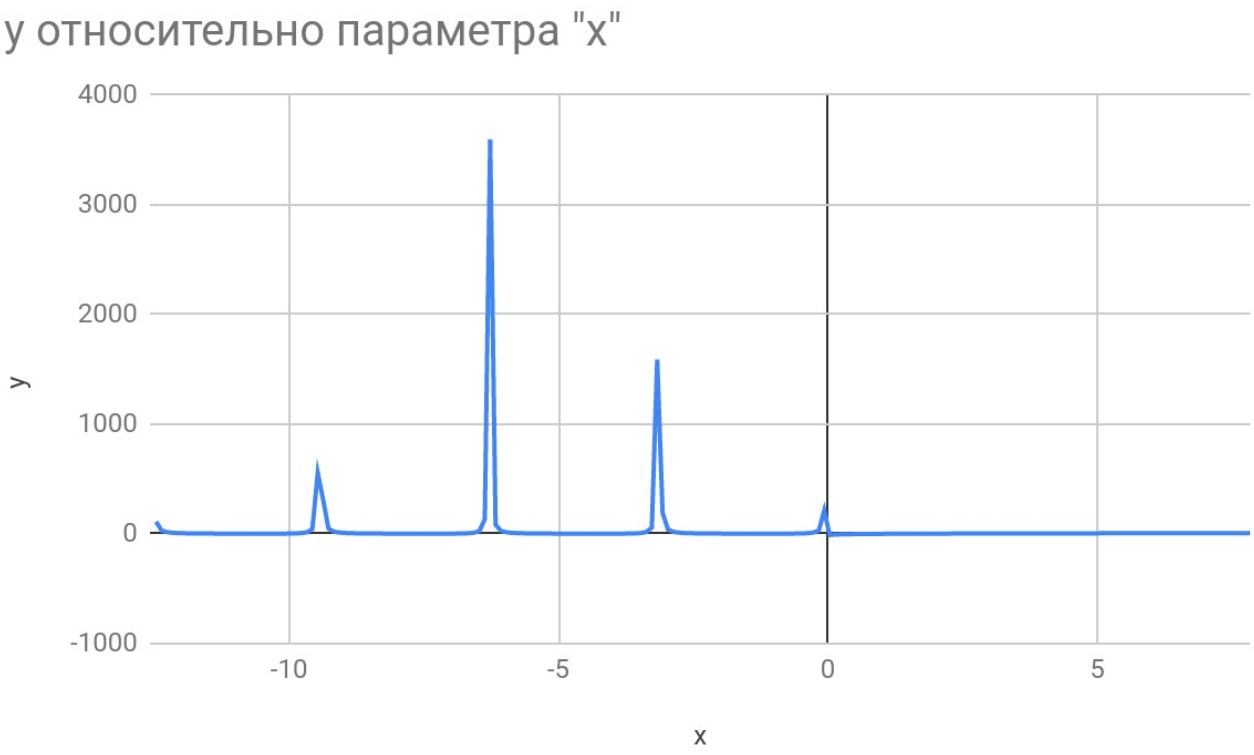
3.Собрать приложение, состоящее из заглушек. Провести интеграцию приложения по 1 модулю, с обоснованием стратегии интеграции, проведением интеграционных тестов и контролем тестового покрытия системы функций.

**UML**

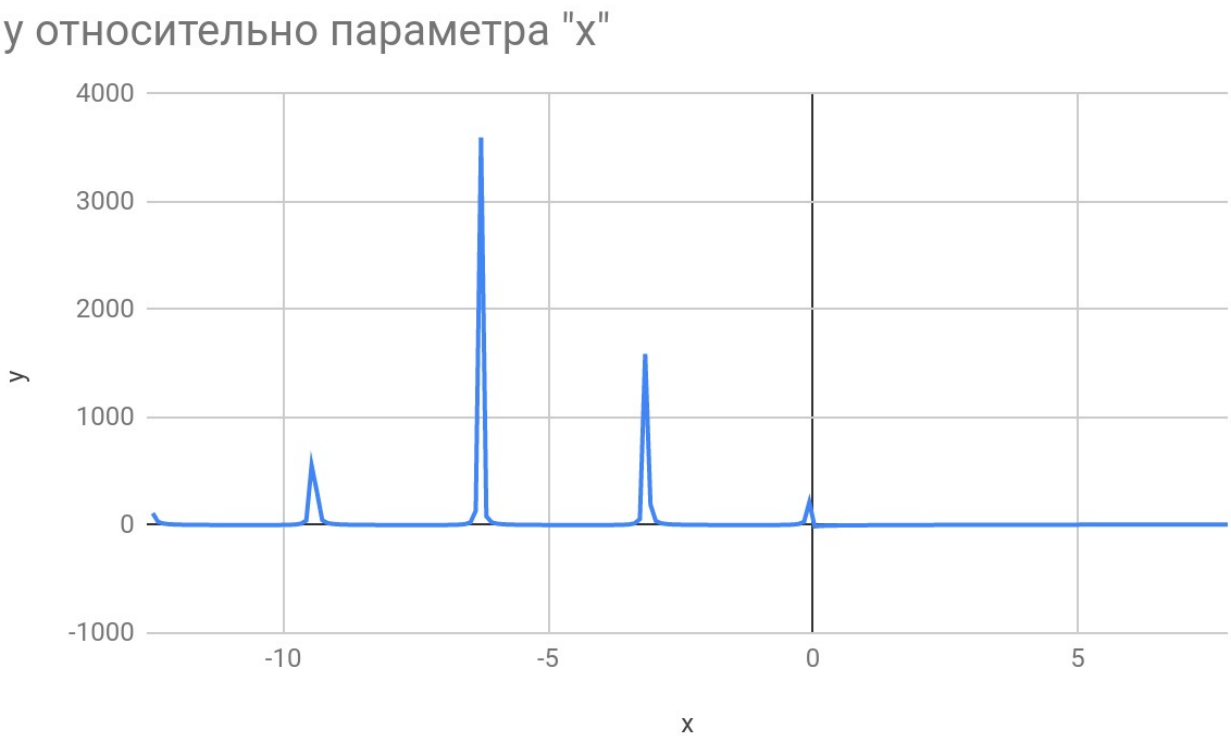


Графики, полученные в ходе интеграции

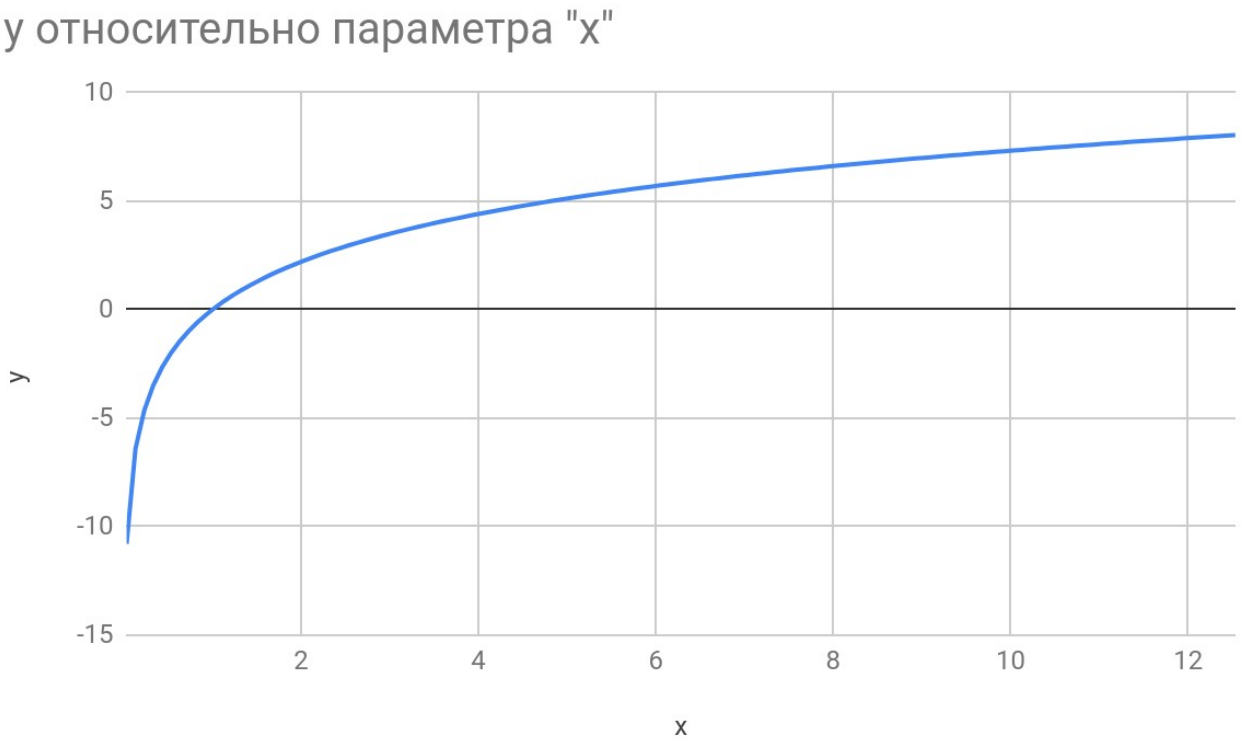
System



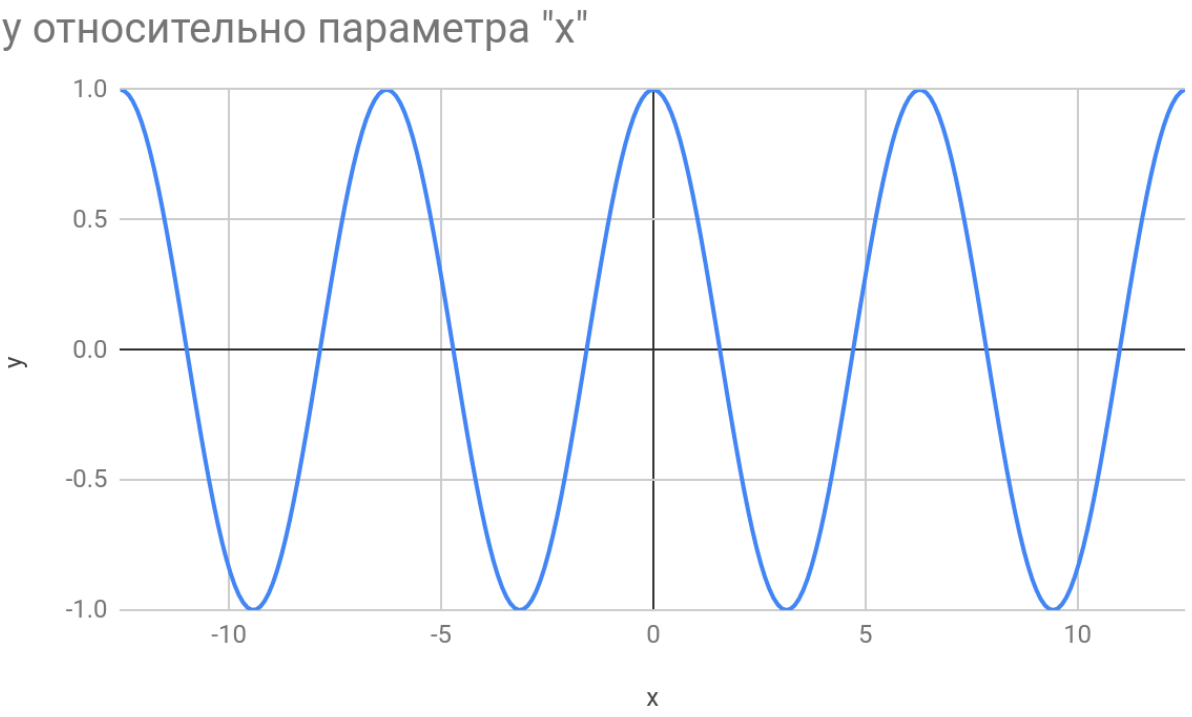

FirstFunction



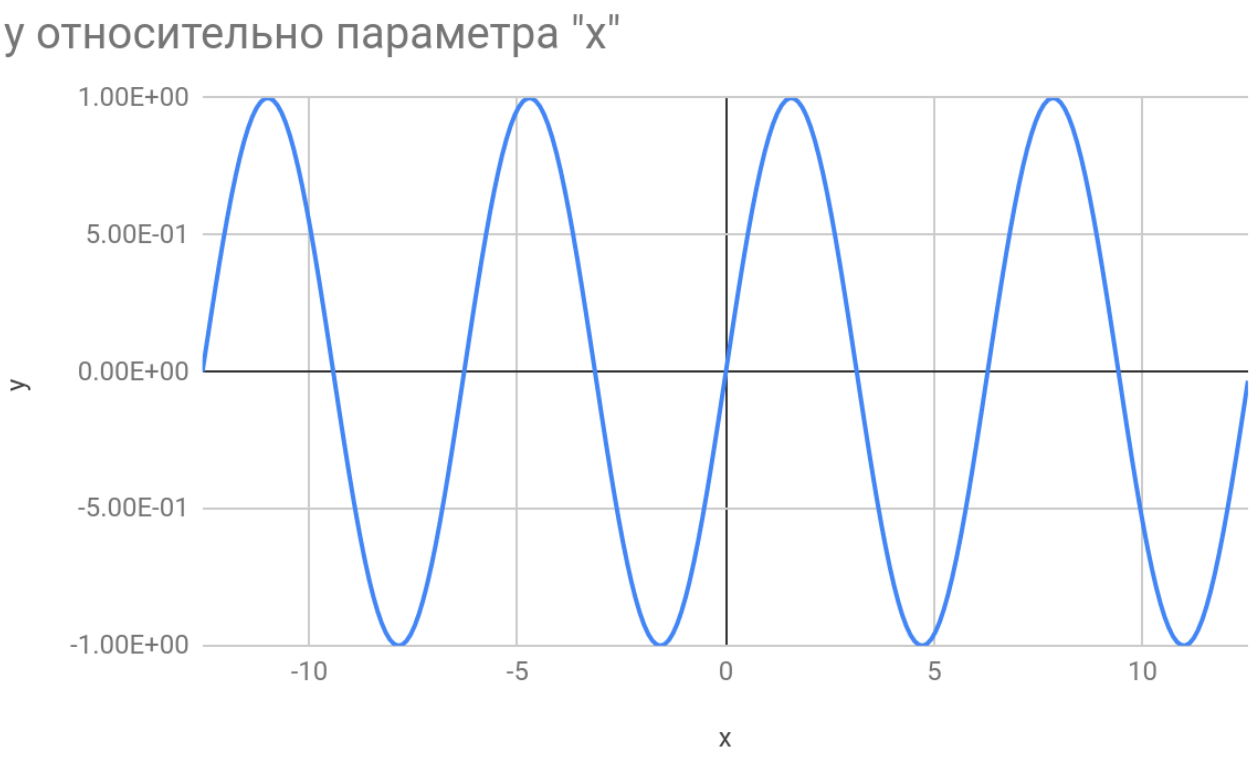
SecondFunction



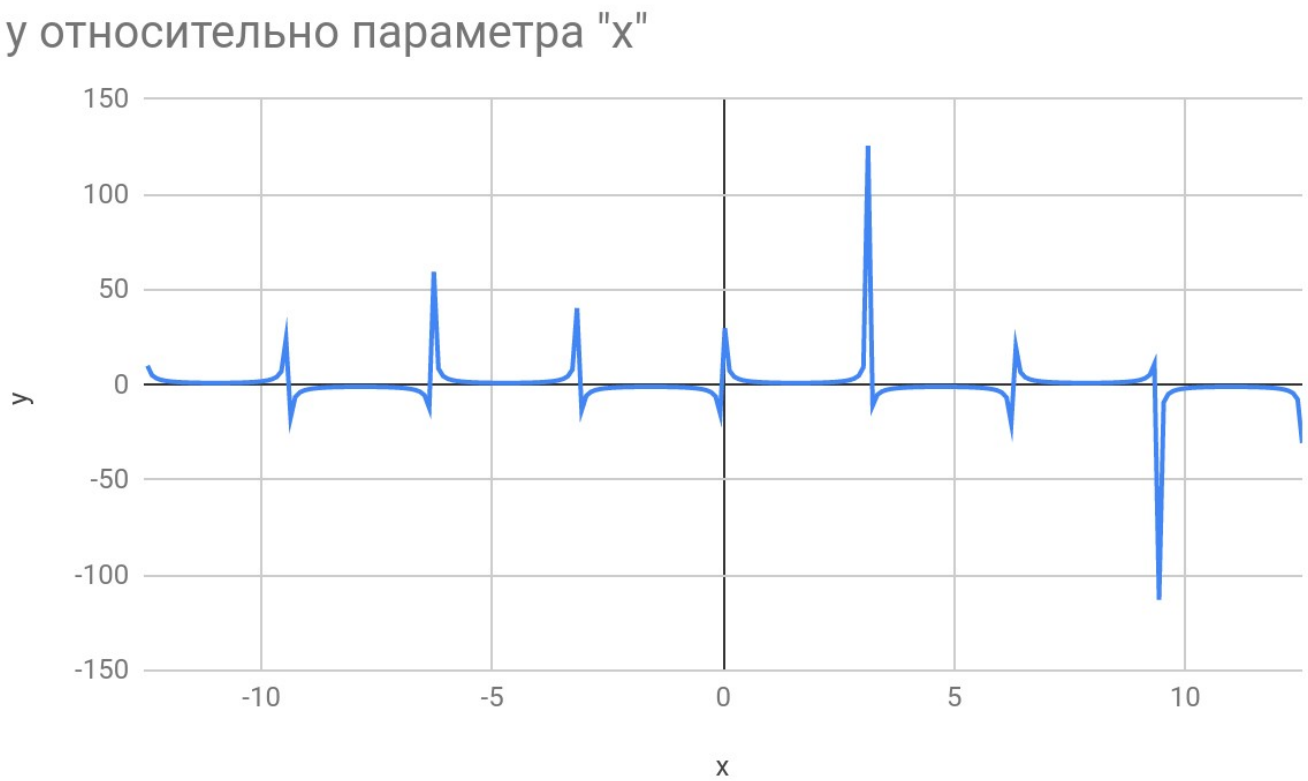

Cosine



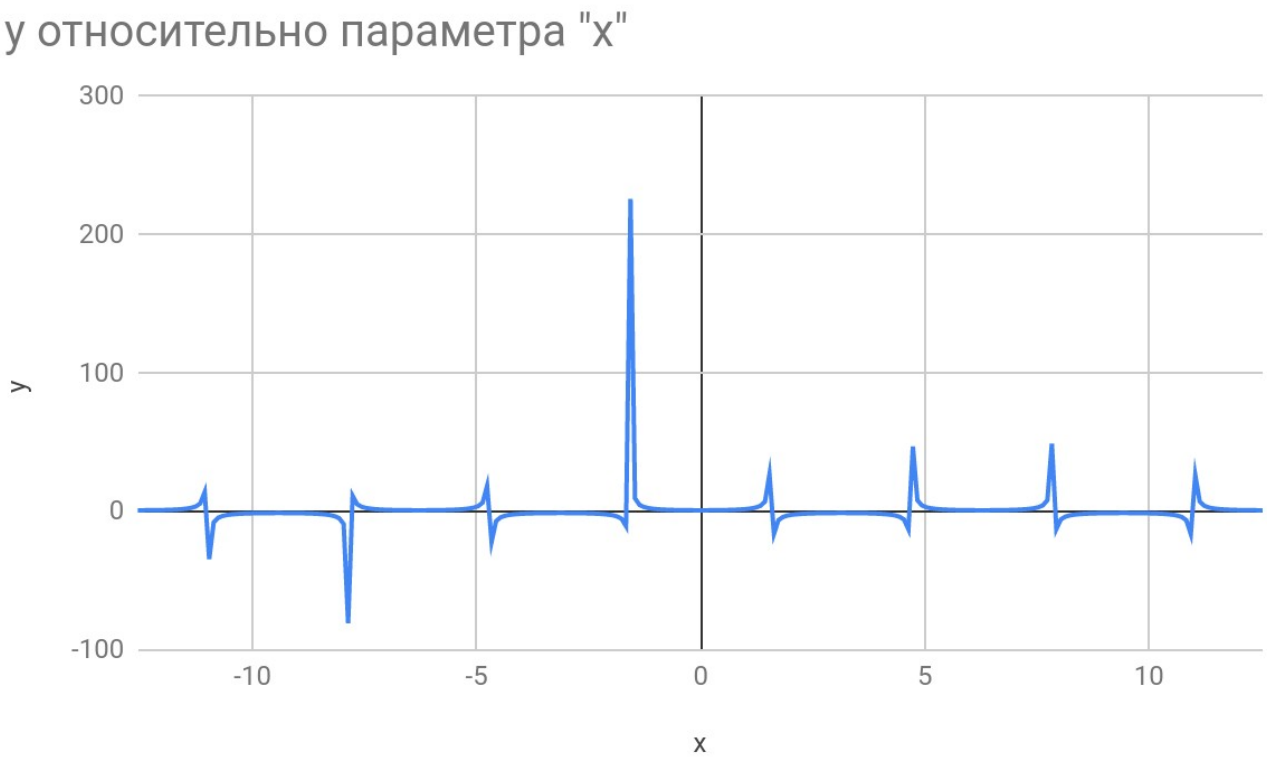
Sinus



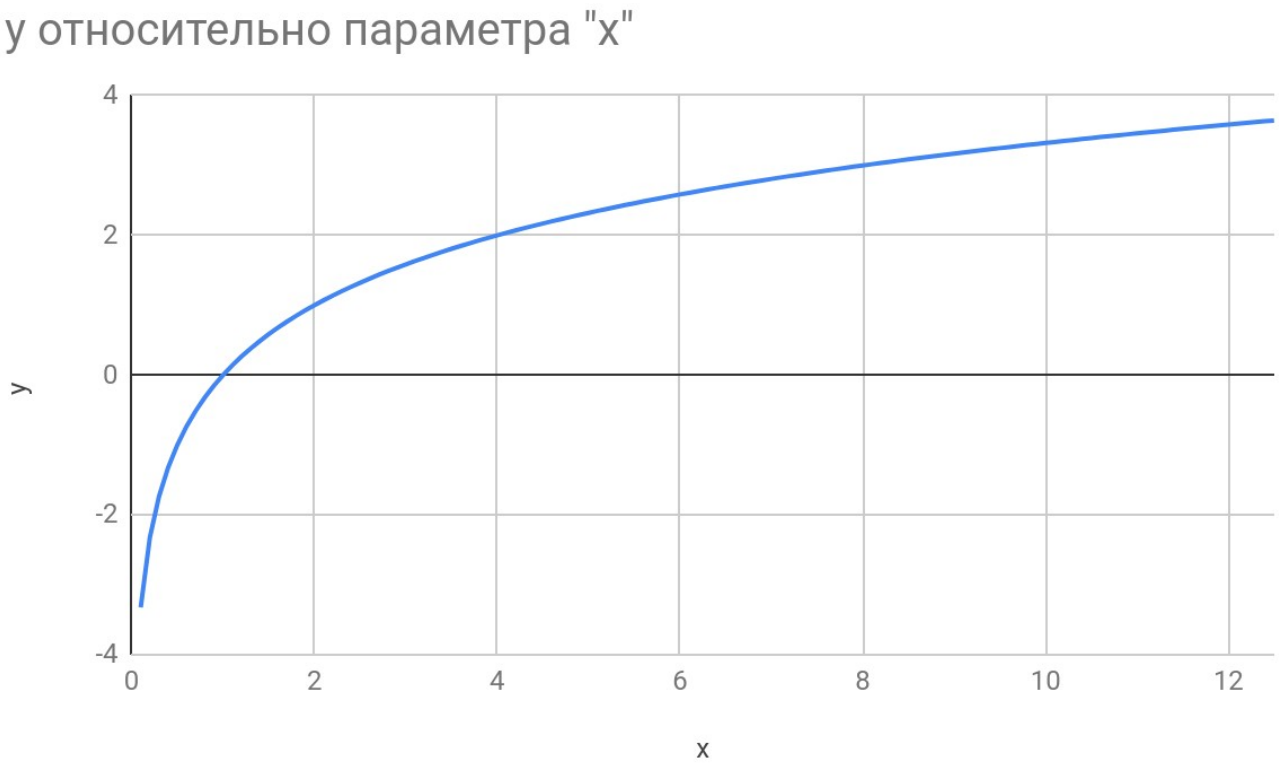

Cosecant



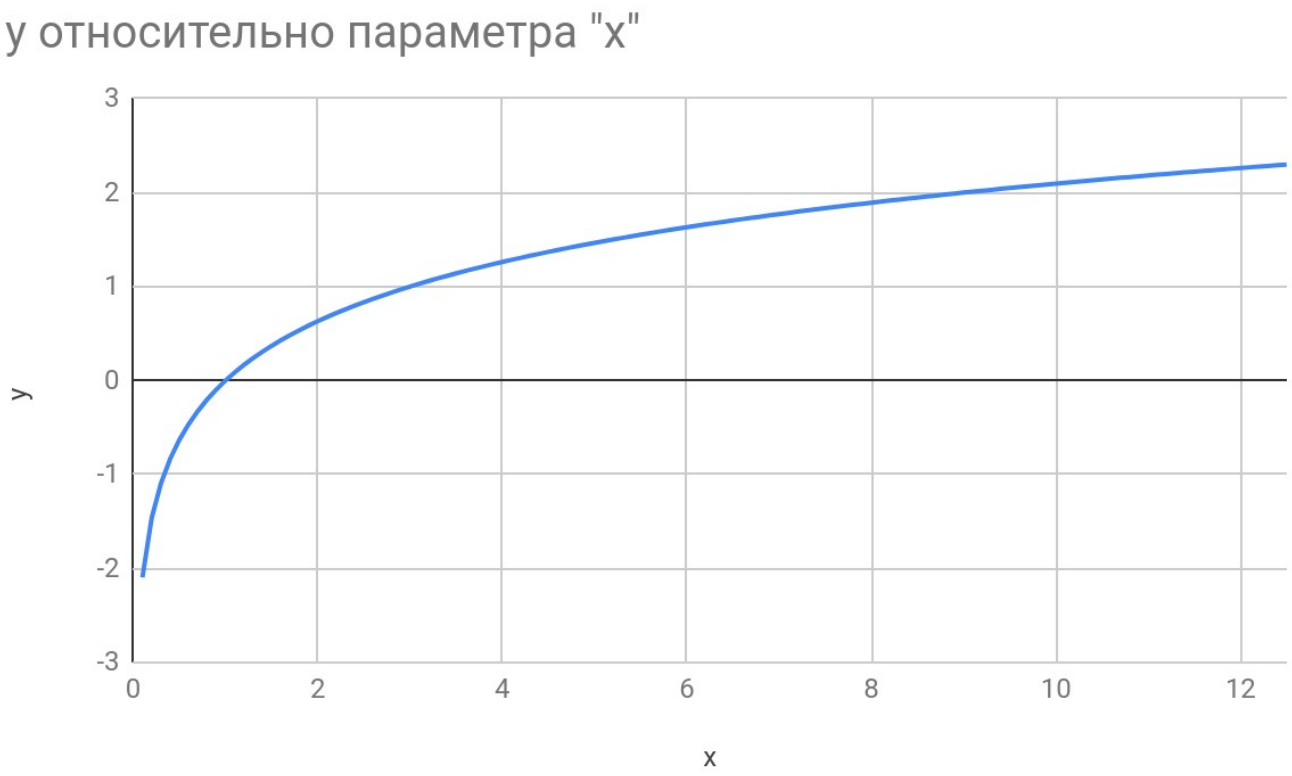
Secant


y относительно параметра "x"
300
200
100
0
-100
y
-10
-5
0
5
10
x

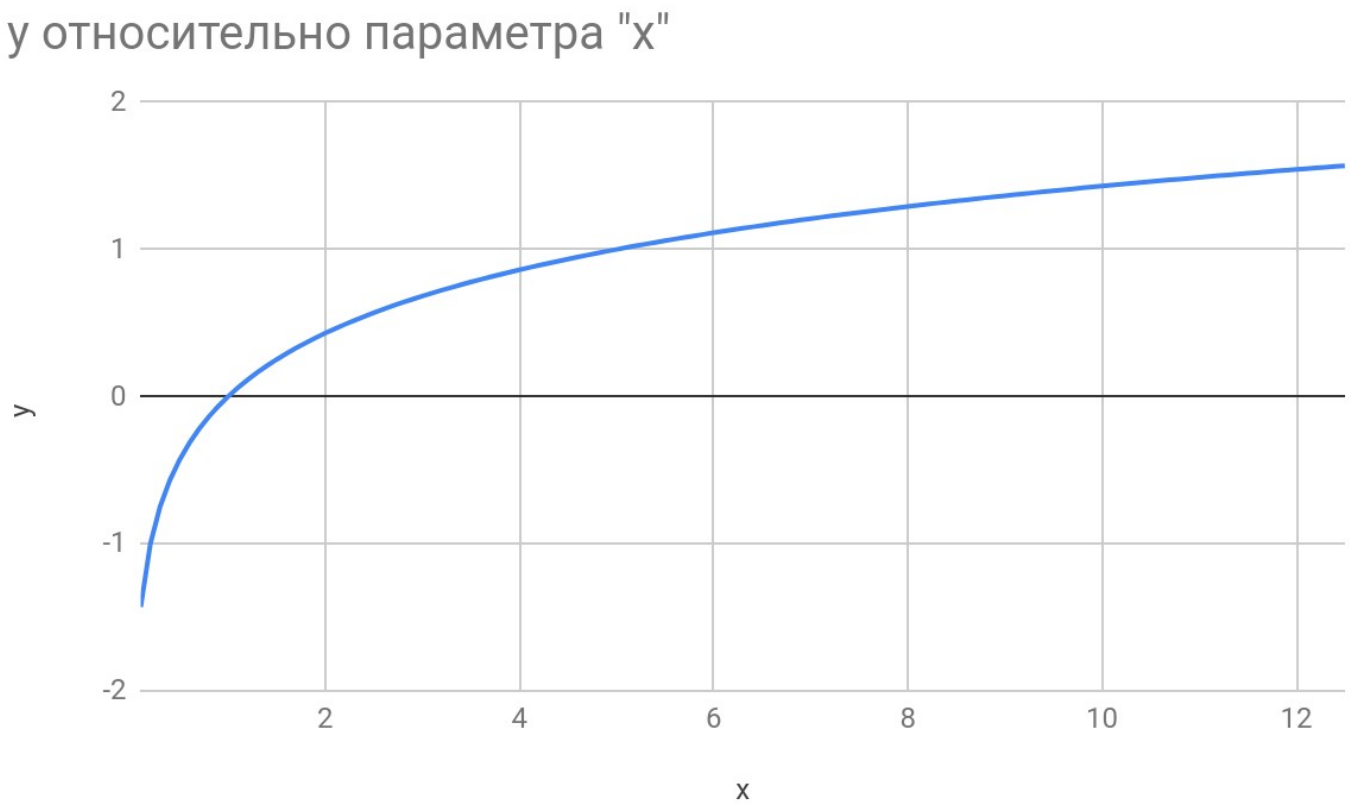
Log2


y относительно параметра "x"
4
2
0
-2
-4
y
0
2
4
6
8
10
12
x

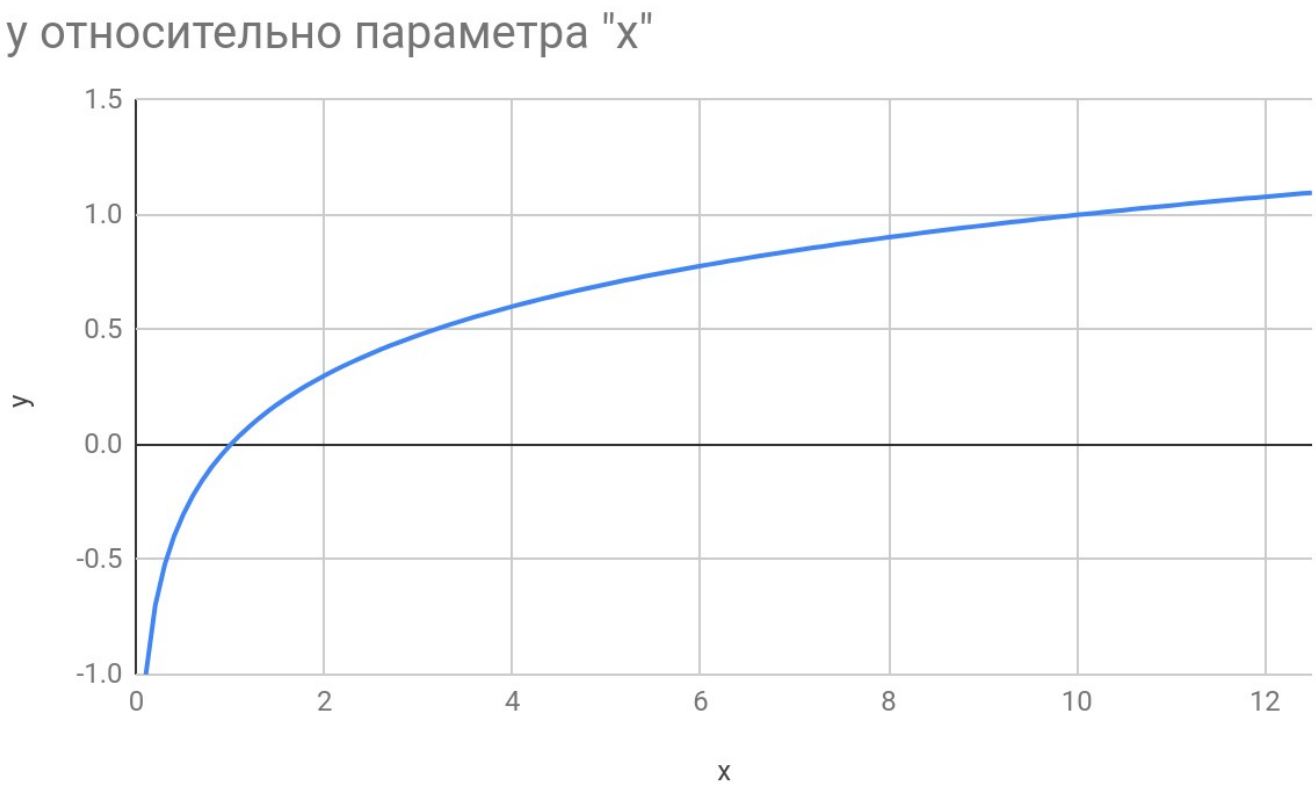
Log3


y относительно параметра "x"
3
2
1
0
-1
-2
-3
y
0
2
4
6
8
10
12
x

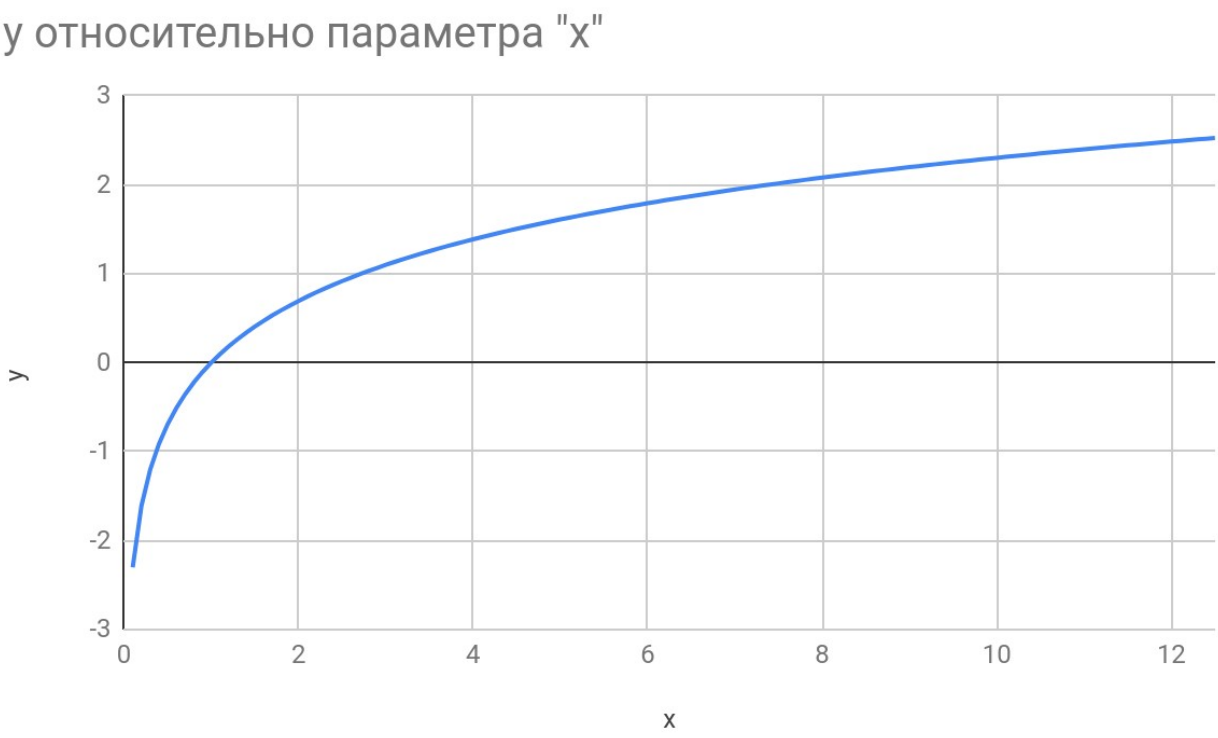
Log5


y относительно параметра "x"
2
1
0
-1
-2
y
2
4
6
8
10
12
x

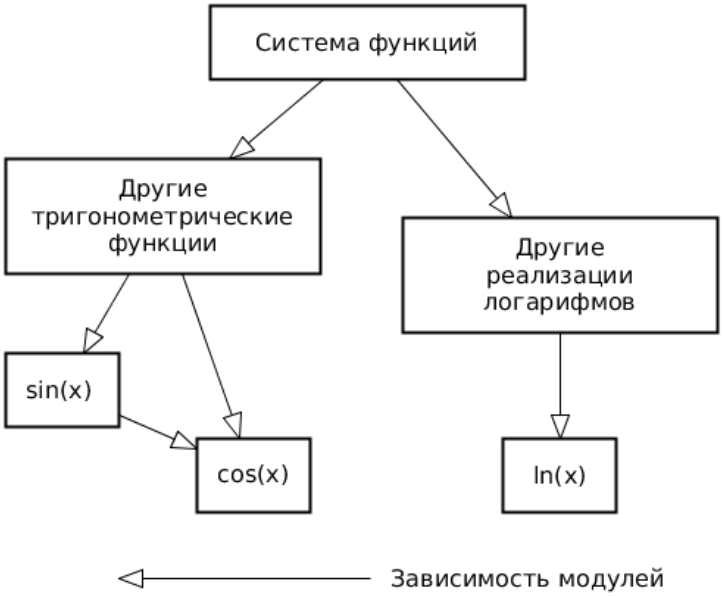
Log10


y относительно параметра "x"
1.5
1.0
0.5
0.0
-0.5
-1.0
y
0
2
4
6
8
10
12
x

LogN

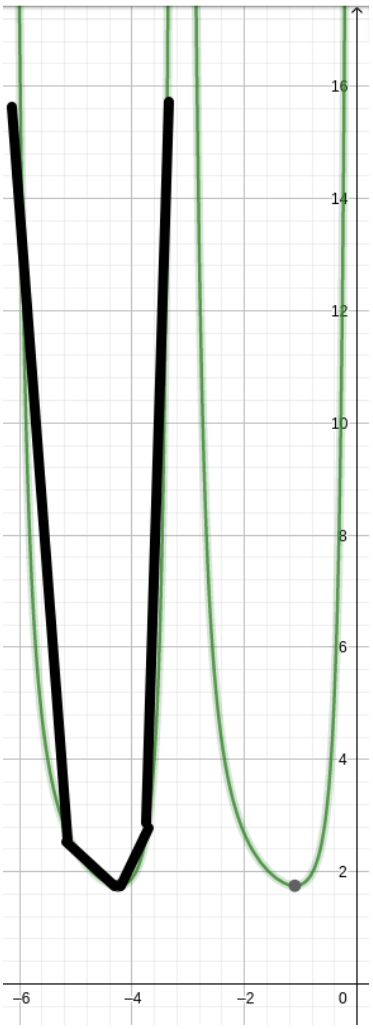

y относительно параметра "x"
3
2
1
0
-1
-2
-3
y
0
2
4
6
8
10
12
x

Стратегия тестирования — сверху вниз.

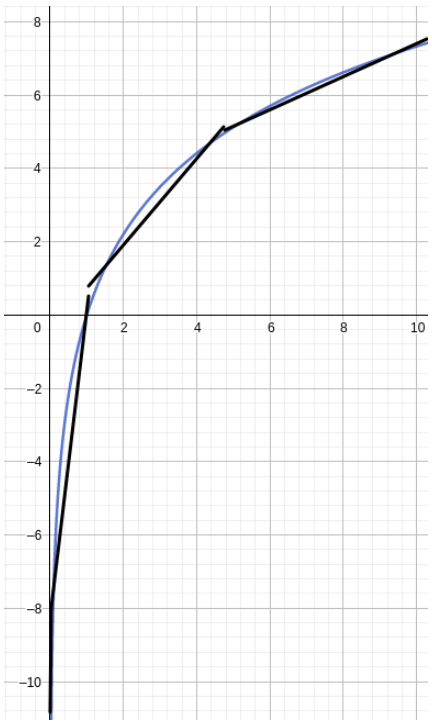



Тестовое покрытие
Для первого этапа — системы, было проведено тестирование на правильное определение выбора функции. Были выбраны значения отрицательные и положительные.

Для последующих этапов каждый отдельный график функций был разбит на партиции эквивалентности, составлено своё тестовое покрытие:
Первая функция:
Было поделено на 4 части одиз из периодов + взят следующий период



Вторая функция:



Код

https://github.com/kevinche75/tpo_lab2_itmo_spring_2021

Вывод: в данной лабораторной работе я изучил интеграционное тестирование, провёл анализ системы функций, составил их тестовое покрытие, разработал интеграционные тесты, познакомился с Mockito.