**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 1**

Тема: Построение изображений 2D-кривых

Студент: Дубровин Дмитрий

Группа: М8О-307Б-21

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2023

1. **Постановка задачи**

Написать и отладить программу, строящую изображение заданной замечательной кривой.

Вариант №16:вставленный-фильм.png

1. **Описание программы**

Для выполнения поставленной задачи было принято решение использовать язык программирования Python и его модули. Этот код создаёт графический пользовательский интерфейс (GUI) с помощью библиотеки Tkinter в Python, который позволяет пользователю вводить значения параметров (a), (A), и (B), и отображает соответствующий график функции, заданной как:вставленный-фильм.png

Описание Элементов:

1. Tkinter Widgets:

- Labels: Для отображения текстовых меток "Введите a (a > 0):", "Введите A (A > -a):", "Введите B (B < a):".

- Entry Widgets: Для ввода значений параметров (a), (A), и (B).

- Buttons: Кнопки "Отрисовать", "Уменьшить" и "Увеличить" для отрисовки графика и изменения масштаба.

2. Matplotlib:

- Используется для создания графика на основе введенных значений и отображения его в пользовательском интерфейсе.

3. Глобальные Переменные:

- `zoom\_factor:` Используется для масштабирования графика при нажатии кнопок увеличенияуменьшения масштаба.

4. Функции:

- `on\_draw():` Отрисовывает график на основе текущих значений параметров и фактора масштабирования.

- `zoom\_in():` Увеличивает масштаб графика на 10% и перерисовывает его.

- `zoom\_out():` Уменьшает масштаб графика на 10% и перерисовывает его.

Рабочий Процесс:

- Пользователь вводит значения параметров (a), (A), и (B).

- При нажатии на кнопку "Отрисовать", программа строит график в соответствии с введенными значениями.

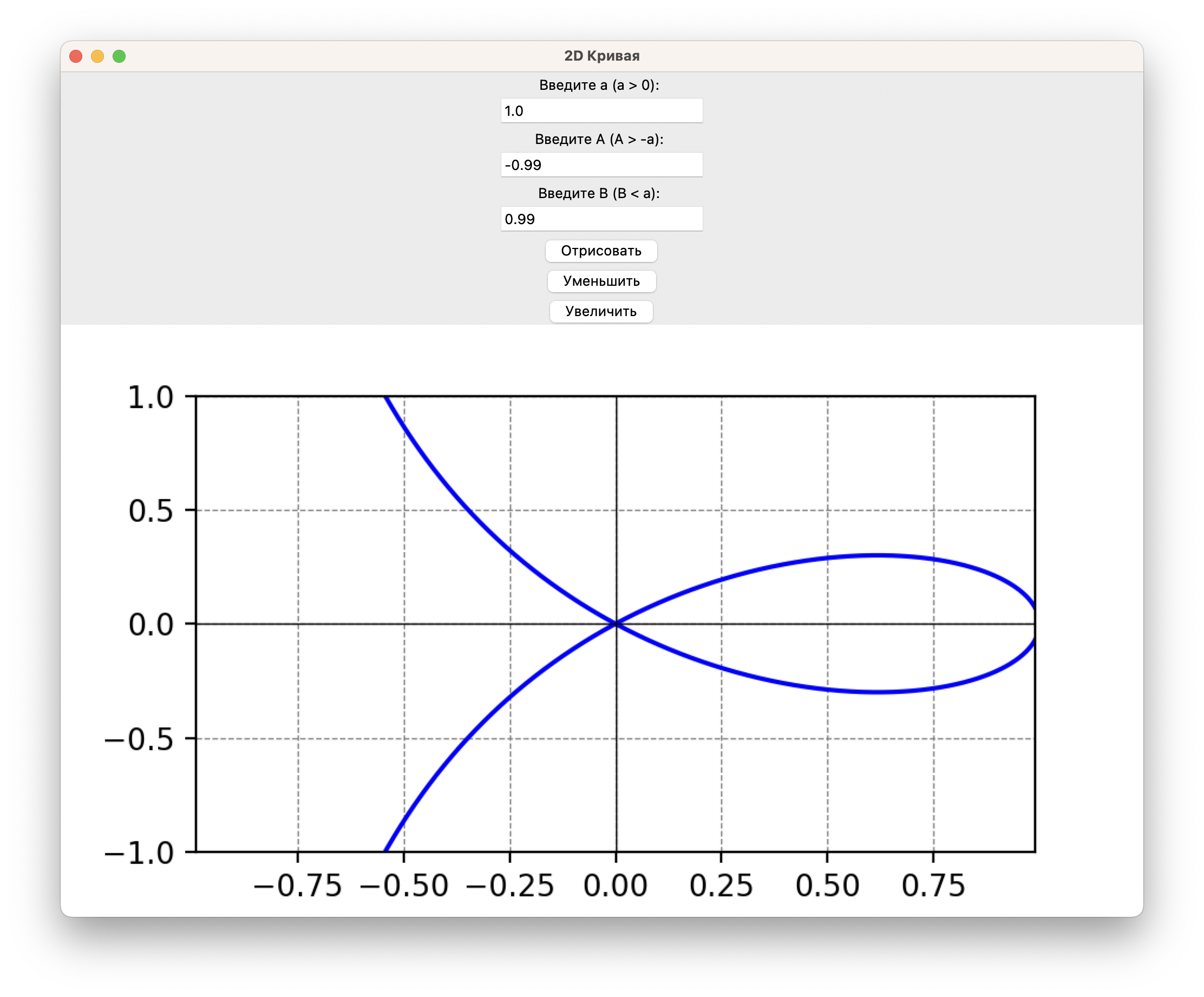
- Пользователь может изменять масштаб графика с использованием кнопок "Увеличить" и "Уменьшить".

- Если введены некорректные значения, программа выводит сообщение об ошибке в консоль.

1. **Результаты выполнения тестов**

a =1.0

A = -0.99

B = 0.99

1. **Листинг программы**

import tkinter as tk

from tkinter import DoubleVar

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

import numpy as np

# Глобальная переменная для фактора масштабирования

zoom\_factor = 1.0

def on\_draw():

global zoom\_factor

# Считываем значение константы a, A и B

a = a\_var.get()

A = A\_var.get()

B = B\_var.get()

# Проверяем, что значения A, B и a корректны

if A >= a or B <= -a or B <= A:

print("Некорректные значения A, B или a")

return

# Создаем массив x в пределах от A до B

x = np.linspace(A, B, 1000)

# Вычисляем y

y\_squared = x\*\*2 \* ((a - x) / (a + x)) # Квадрат y

y\_squared = np.maximum(y\_squared, 0) # Убедимся, что значения неотрицательны

y = np.sqrt(y\_squared) # Вычислим корень для получения y

# Обновляем график с учетом масштабирования

ax.cla()

ax.plot(x, y, 'b')

ax.plot(x, -y, 'b')

ax.axhline(0, color='black', linewidth=0.5)

ax.axvline(0, color='black', linewidth=0.5)

ax.grid(color='gray', linestyle='--', linewidth=0.5)

# Устанавливаем новые пределы осей

ax.set\_xlim(A \* zoom\_factor, B \* zoom\_factor)

ax.set\_ylim(-a \* zoom\_factor, a \* zoom\_factor)

canvas.draw()

def zoom\_in():

global zoom\_factor

zoom\_factor \*= 1.1 # Увеличиваем фактор масштабирования на 10%

on\_draw() # Перерисовываем график после зума

def zoom\_out():

global zoom\_factor

zoom\_factor /= 1.1 # Уменьшаем фактор масштабирования на 10%

on\_draw() # Перерисовываем график после зума

root = tk.Tk()

root.title("2D Кривая")

# Создаем tkinter переменные для констант a, A и B

a\_var = DoubleVar(value=1.0)

A\_var = DoubleVar(value=-0.99)

B\_var = DoubleVar(value=0.99)

# Создаем поля для ввода констант a, A и B

a\_label = tk.Label(root, text="Введите a (a > 0): ")

a\_label.pack()

a\_entry = tk.Entry(root, textvariable=a\_var)

a\_entry.pack()

A\_label = tk.Label(root, text="Введите A (A > -a): ")

A\_label.pack()

A\_entry = tk.Entry(root, textvariable=A\_var)

A\_entry.pack()

B\_label = tk.Label(root, text="Введите B (B < a): ")

B\_label.pack()

B\_entry = tk.Entry(root, textvariable=B\_var)

B\_entry.pack()

# Создаем кнопки для отрисовки и управления масштабом

draw\_button = tk.Button(root, text="Отрисовать", command=on\_draw)

draw\_button.pack()

zoom\_in\_button = tk.Button(root, text="Уменьшить", command=zoom\_in)

zoom\_in\_button.pack()

zoom\_out\_button = tk.Button(root, text="Увеличить", command=zoom\_out)

zoom\_out\_button.pack()

# Настраиваем фигуру и область рисования

fig, ax = plt.subplots(figsize=(5, 5))

canvas = FigureCanvasTkAgg(fig, master=root)

canvas.get\_tk\_widget().pack()

# Запускаем главный цикл tkinter

root.mainloop()

1. **Вывод**

В ходе данной лабораторной работы я изучил несколько полезных библиотек Python, с которыми только пересекался ранее. Лабораторная работа была успешно выполнена, все поставленные задачи были решены, а цели достигнуты. Программа демонстрирует корректную работу и предоставляет пользователям возможность визуализации графика на основе введенных параметров, а также проведения анализа графика путем изменения масштаба. Эта работа демонстрирует, как можно сочетать графический интерфейс и визуализацию данных для создания полезных и эффективных инструментов анализа.

**Литература**

Numpy documentation [Электронный ресурс] URL: <https://numpy.org>

Matplotlib tutorials [Электронный ресурс] URL: [https://matplotlib.org](https://matplotlib.org/)

Tkinter documentation [Электронный ресурс] URL: <https://docs.python.org/3/library/tkinter.html>