Лабораторная работа №2

**Тема:** Каркасная визуализация выпуклого многогранника. Удаление невидимых линий.

**Задача:** Разработать формат представления многогранника и процедуру его каркасной отрисовки в ортографической и изометрической проекциях. Обеспечить удаление невидимых линий и возможность пространственных поворотов и масштабирования многогранника. Обеспечить автоматическое центрирование и изменение размеров изображения при изменении размеров окна.

**Вариант №16: гранная прямая правильная пирамида**

1. Решение

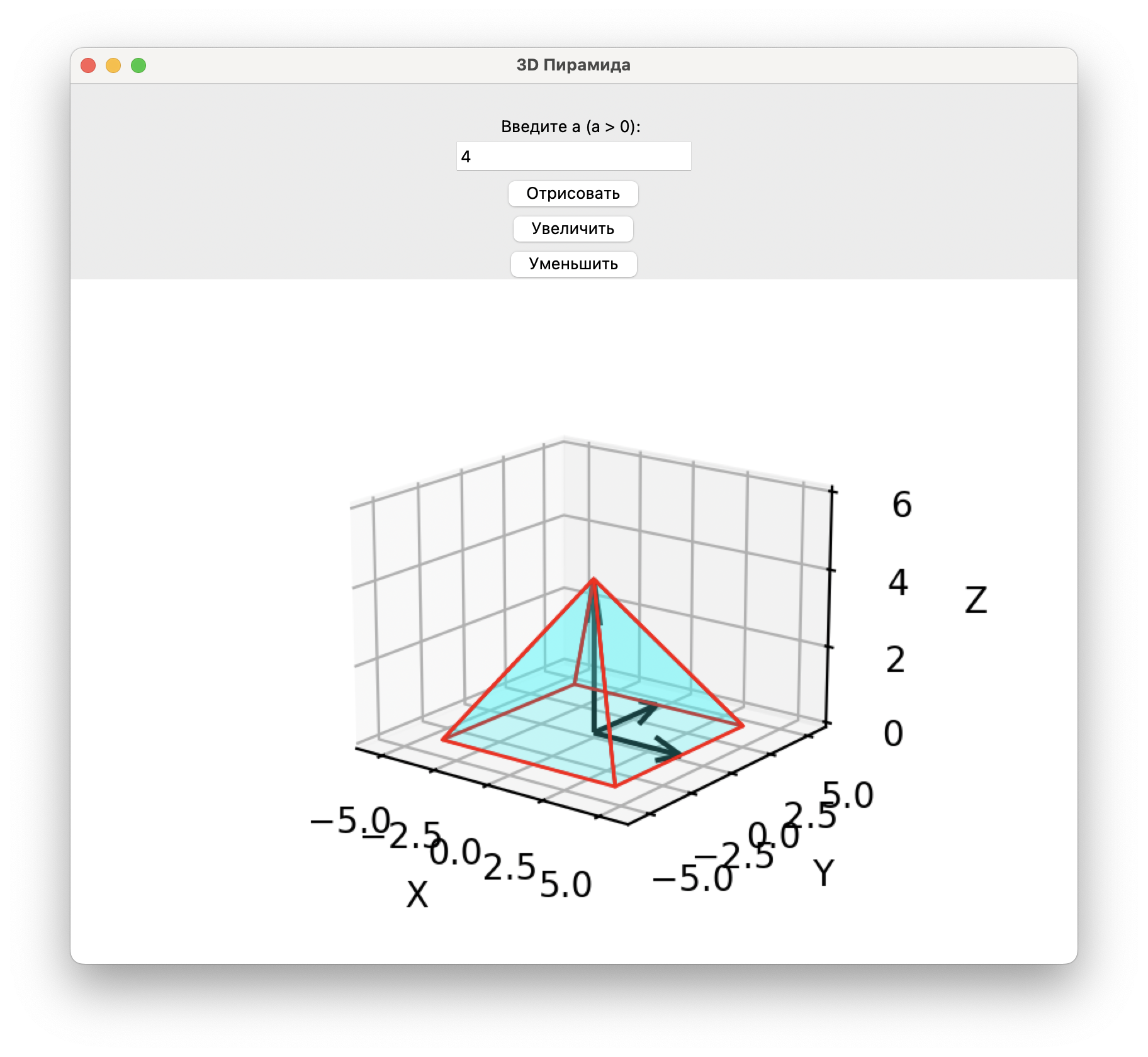
Для выполнения поставленной задачи было принято решение использовать

язык программирования Python и его модули. Этот код создаёт графический

пользовательский интерфейс (GUI) с помощью библиотеки Tkinter в Python,

который позволяет пользователю вводить значения параметра а - для пирамид с различной длинной ребра.

Давайте рассмотрим каждую функцию программы по порядку:

1. center\_window(window, width, height): Эта функция центрирует окно графического интерфейса на экране. Она принимает параметры window (окно Tkinter), width (ширина окна) и height (высота окна), и использует их для вычисления координат для размещения окна в центре экрана.
2. on\_draw(): Эта функция вызывается при нажатии кнопки "Отрисовать". Она выполняет следующие действия:
   * Считывает значение константы a из виджета a\_var.
   * Проверяет, что значение a больше нуля. Если a меньше или равно нулю, выводит сообщение об ошибке в виджет error\_label и завершает выполнение функции.
   * Вычисляет координаты вершин пирамиды на основе значения a.
   * Создает вершины и грани пирамиды.
   * Рисует трехмерную пирамиду и стрелки на осях в окне с помощью Matplotlib.
3. zoom\_in(): Эта функция вызывается при нажатии кнопки "Увеличить". Она увеличивает масштаб пирамиды на 10% относительно текущего масштаба, но только если текущий масштаб (значение zoom\_factor) больше или равен 0.69 (чтобы предотвратить слишком большое увеличение). Затем она вызывает функцию on\_draw() для перерисовки пирамиды с новым масштабом.
4. zoom\_out(): Эта функция вызывается при нажатии кнопки "Уменьшить". Она уменьшает масштаб пирамиды на 10% относительно текущего масштаба, но только если текущий масштаб больше или равен 0.69 (чтобы предотвратить слишком большое уменьшение). Затем она вызывает функцию on\_draw() для перерисовки пирамиды с новым масштабом.
5. Создание элементов интерфейса:
   * a\_label и a\_entry: Это метка и поле ввода для значения a.
   * draw\_button: Кнопка "Отрисовать" для запуска функции on\_draw().
   * zoom\_in\_button и zoom\_out\_button: Кнопки "Увеличить" и "Уменьшить" для управления масштабом пирамиды.
6. Создание трехмерной фигуры с помощью Matplotlib:
   * fig: Создание объекта Figure (фигуры) Matplotlib.
   * ax: Добавление подграфика с трехмерной проекцией (projection='3d') к фигуре. На этом подграфике будет отображаться пирамида.
   * canvas: Создание объекта FigureCanvasTkAgg для вставки фигуры Matplotlib в окно Tkinter.
7. Главный цикл Tkinter root.mainloop(): Этот цикл запускает главное окно и обеспечивает его работу, позволяя пользователю взаимодействовать с интерфейсом.
8. 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 фигуры на основе введенных параметров, а также проведения анализа фигуры путем изменения масштаба. Эта работа демонстрирует, как можно сочетать графический интерфейс и визуализацию данных для создания эффективных инструментов анализа.