МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра САПР

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Компьютерная графика»

Тема: Поворот объемного тела относительно осей координат на заданный угол.

Студенты гр. 1371 _____ Вотяков Ф.А., Гореликов М.А., Кочетков А.С.

Преподаватель _____ Матвеева И.В.

Санкт-Петербург

2024

ЗАДАНИЕ

Разработать программу, обеспечивающую поворот объемного тела относительно осей координат на заданный угол.

ТЕОРИТИЧЕСКИЕ ПОЛОЖЕНИЯ

Для обеспечения поворота объемного тела вокруг оси необходимо воспользоваться матрицей преобразования

$$[X][T]=[x \quad y \quad z \quad 1]\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & e & 0 & 0 \\ 0 & 0 & j & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}=[ax \quad ey \quad zj \quad 1]=[x^* \quad y^* \quad z^* \quad 1]$$

Рисунок 1Формула преобразования координат в пространстве

Вид матрицы поворота для каждой из осей имеют вид:

Поворот фигуры вокруг оси ‘z’

$$T_z=\left[\begin{array}{ccc|c} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Поворот фигуры вокруг оси ‘x’

$$T_x=\left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & \cos\beta & \sin\beta & 0 \\ 0 & -\sin\beta & \cos\beta & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Поворот фигуры вокруг оси ‘y’

$$T_y=\left[\begin{array}{ccc|c} \cos\gamma & 0 & -\sin\gamma & 0 \\ 0 & 1 & 0 & 0 \\ \sin\gamma & 0 & \cos\gamma & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

Рисунок 2 Матрицы преобразования для поворота трехмерной фигуры

Для перемножения матриц необходимо добавить 1 в конец каждого вектора-строки [x,y,z] => [x,y,z,1]

- Преобразование выполняется в правосторонней системе координат.
- Вращается сам объект
- Положительный поворот определяется правилом правой руки.
- Координаты записаны в виде строки матрицы
- Ось, вокруг которой будет осуществляться поворот можно выбрать

ОПИСАНИЕ ИНТЕРЕЙСА

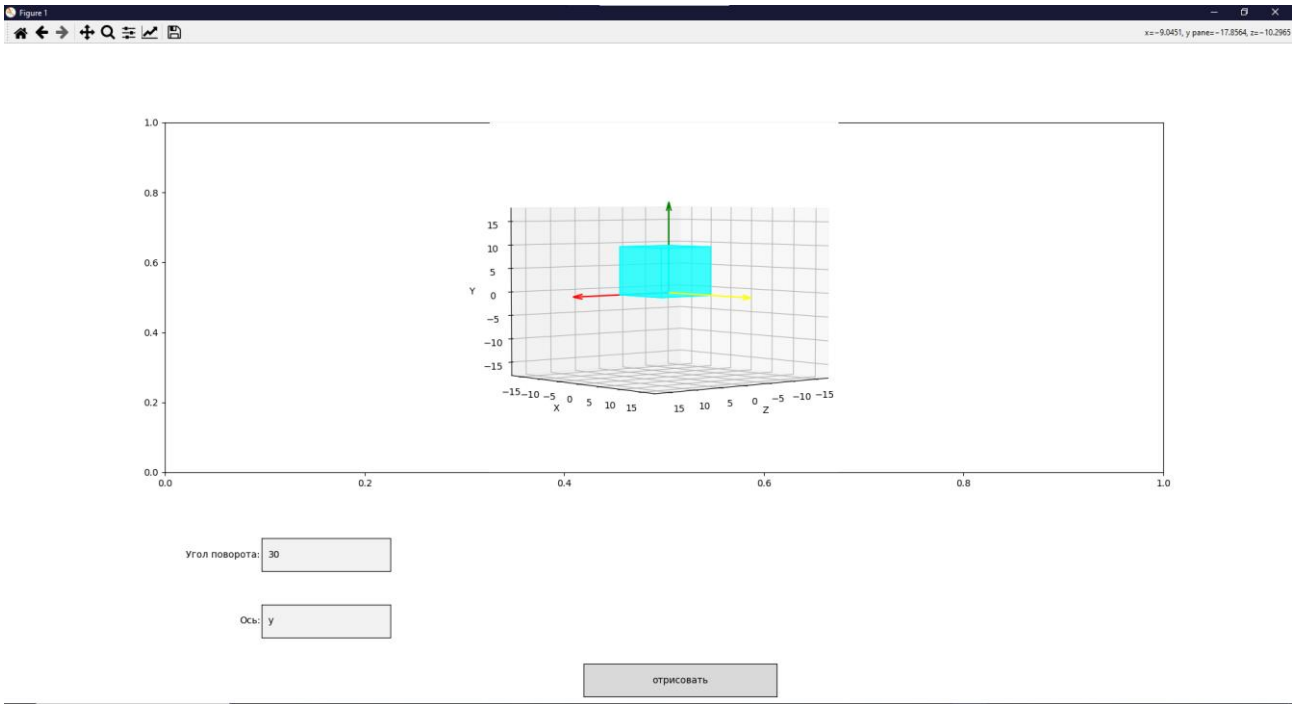

Рисунок 3 стандратное отображение интерфейса

На экране изображаются координатные оси (z – красным цветом, y – зеленым, x – желтым) для удобства просмотра
Поле ввода «Угол поворота:» позволяет ввести угол на который будет повернута фигура, в градусах, можно как положительное, так и отрицательное целое число
Поле ввода «Ось:» принимает одну из букв: z/x/y и отвечает за ось вокруг которой будет осуществлен поворот
Кнопка «Отрисовать» по нажатию создает новый график с исходным кубом и его повернутой копией

РАБОТА ПРОГРАММЫ

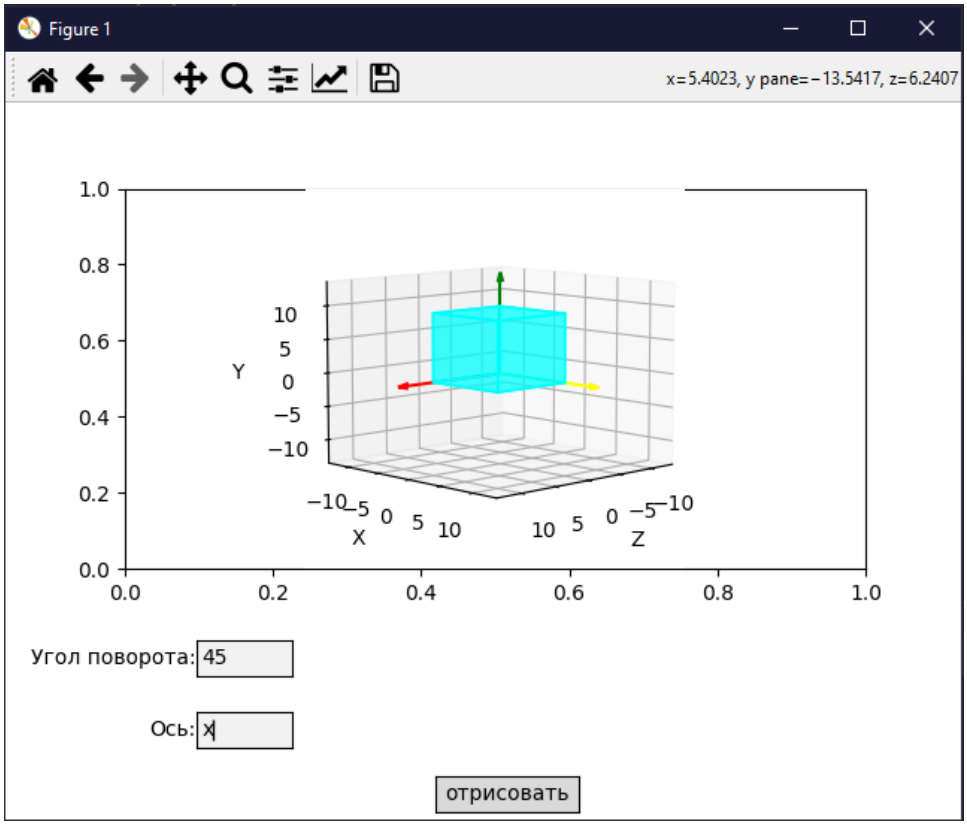

Рисунок 4 изначальное положение куба

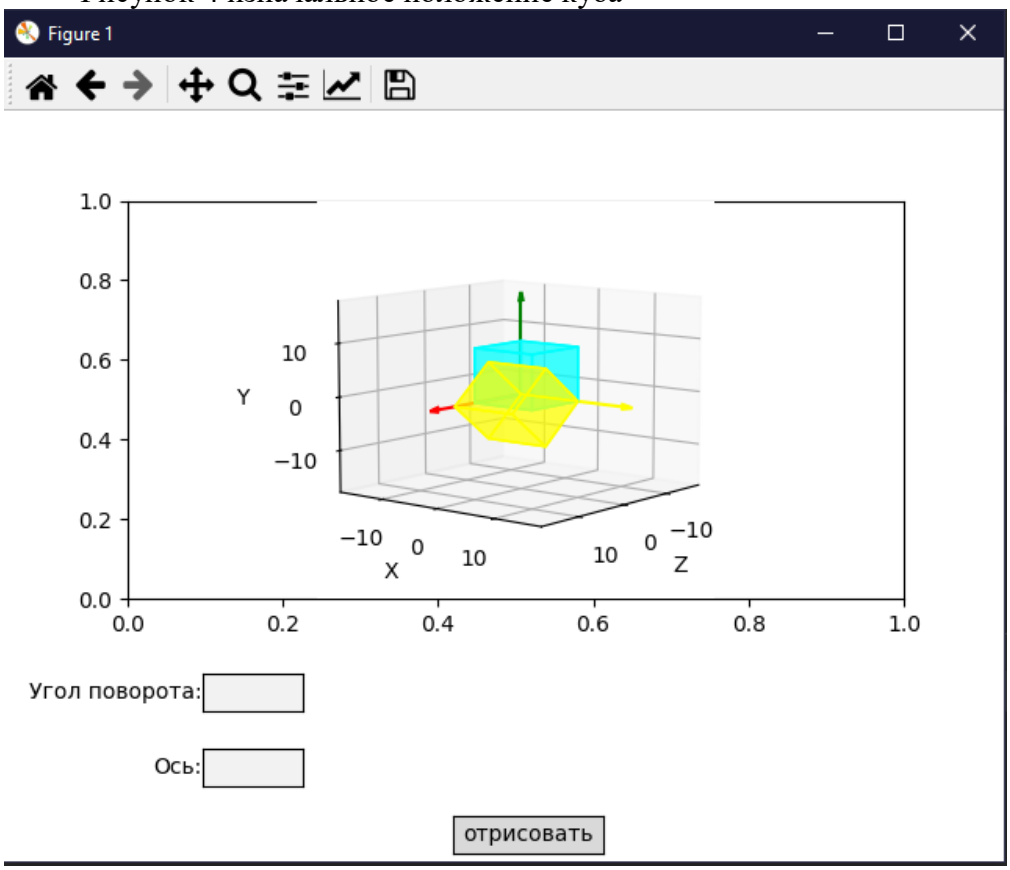

Рисунок 5отображение куба, повернутого на 45 градусов по оси x

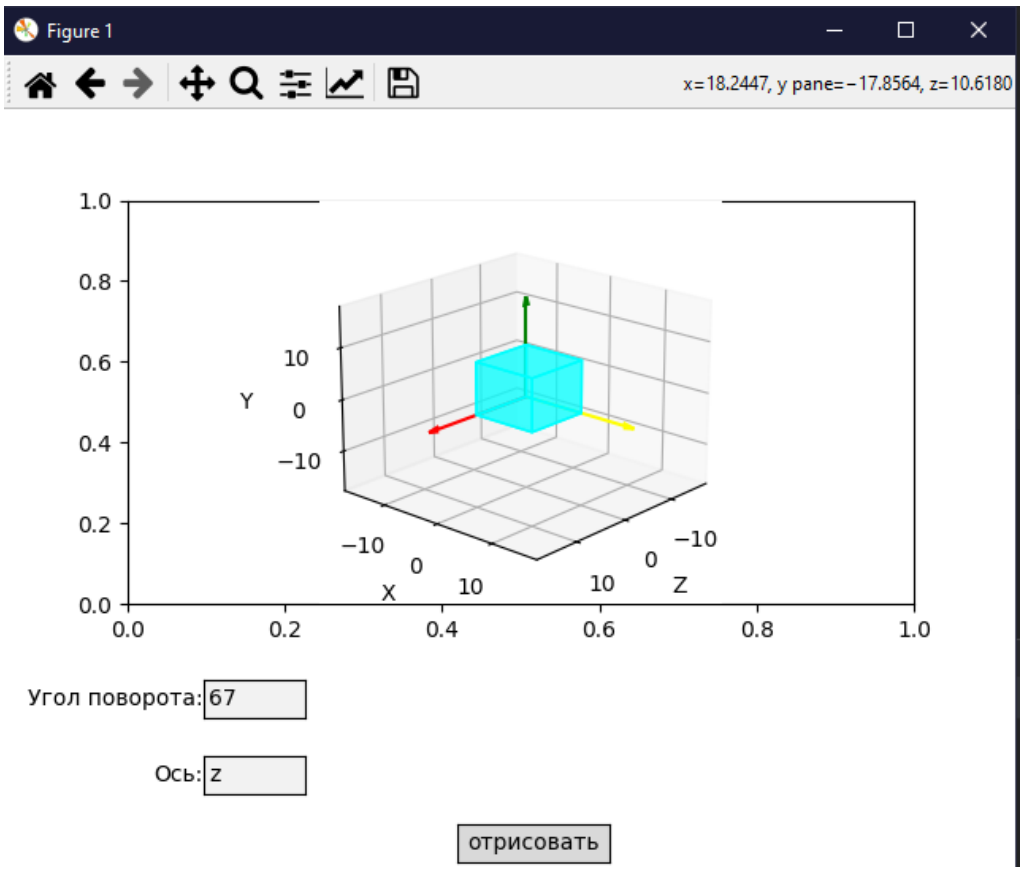

Рисунок 6 изначальное положение куба

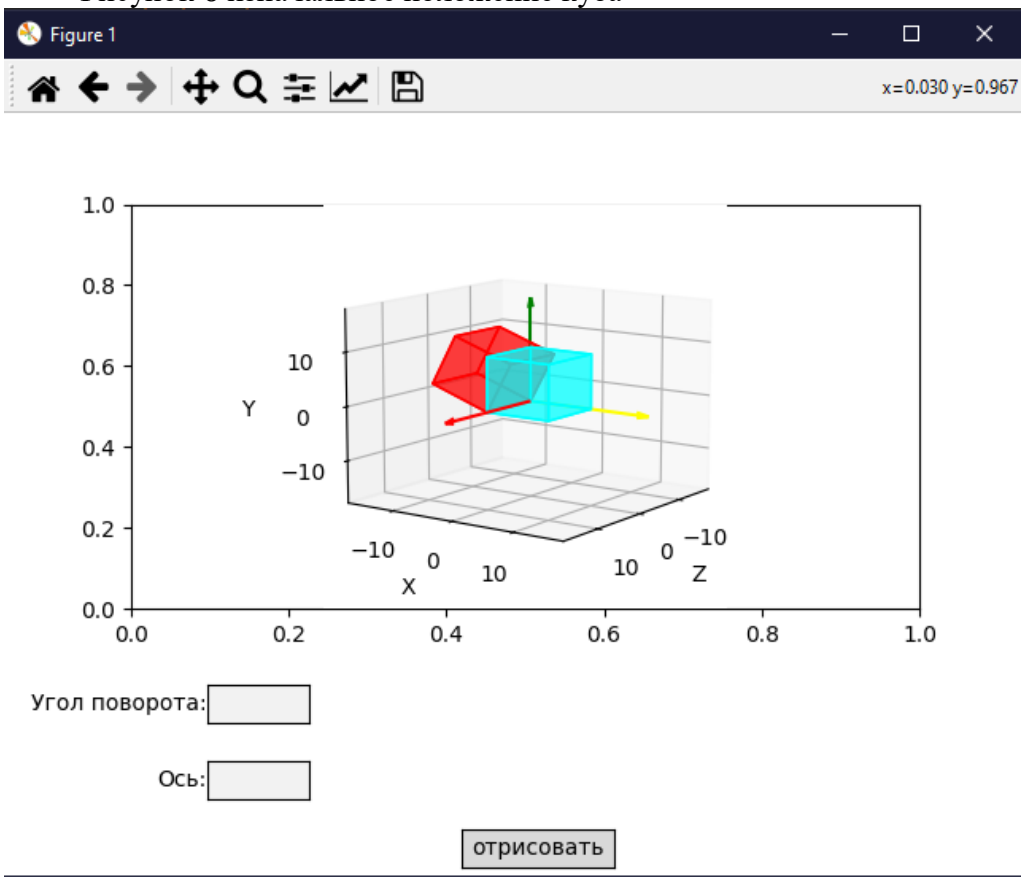

Рисунок 7 поворот на 67 градусов по оси z

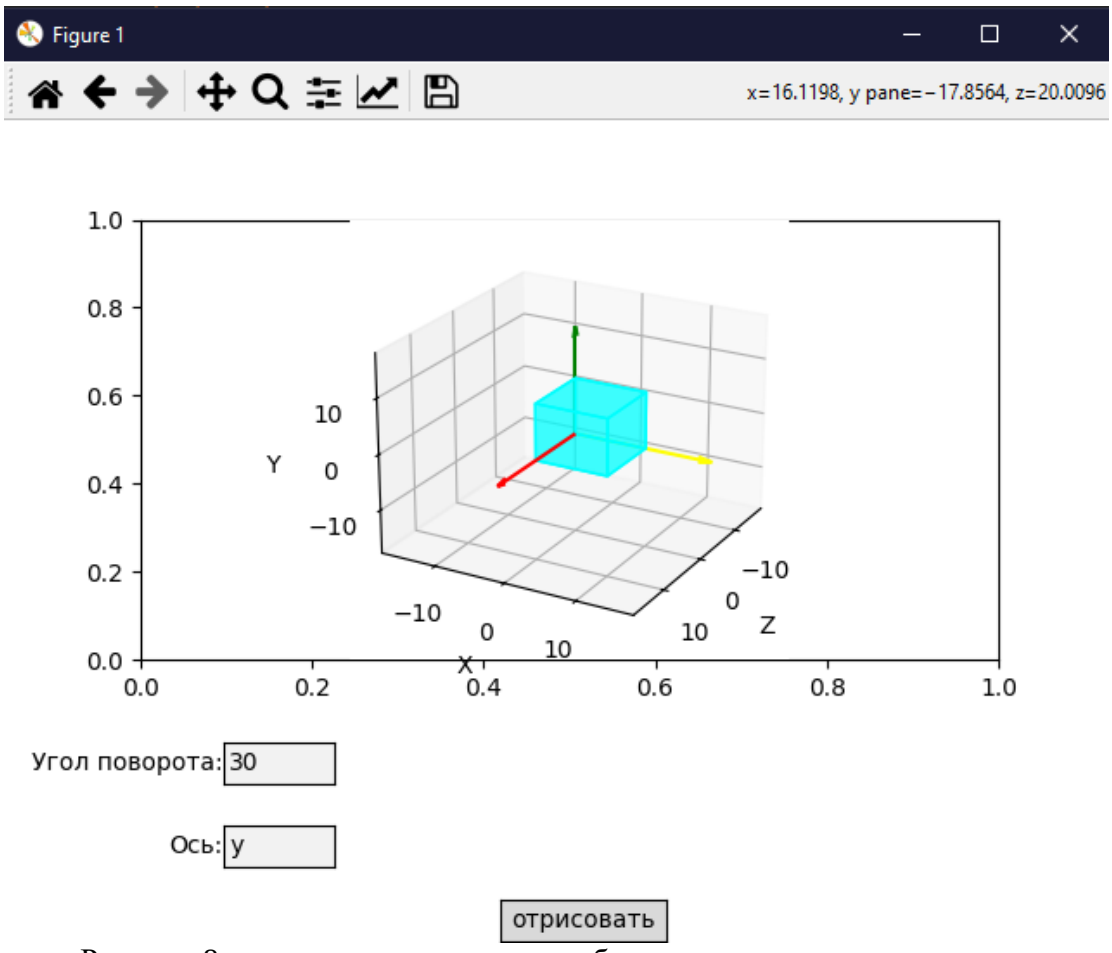

Рисунок 8 изначальное положение куба

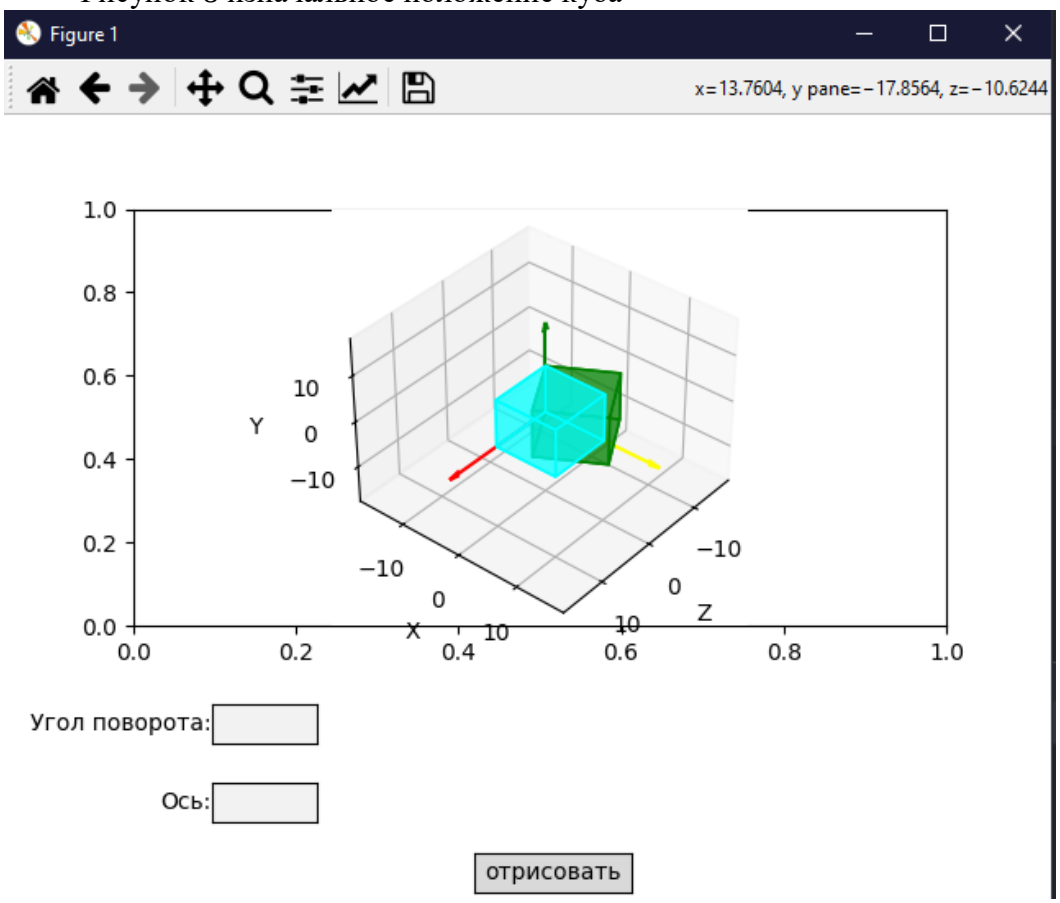

Рисунок 9 поворот на 30 градуса по оси y

ПРОГРАММНЫЙ КОД

Метод поворота куба

```
    def rotate(self, angel, axis):
        """angel указывается числом в градусах
           axis указывается строкой из одного символа \"x\",\"z\,\"y\"
           """

        angel = radians(angel)
        self.add1()
        if axis == "z":
            self.rotate_matrix = [[cos(angel), sin(angel), 0, 0],
                                  [-1 * sin(angel), cos(angel), 0, 0],
                                  [0, 0, 1, 0],
                                  [0, 0, 0, 1]]
            color = "green"
        elif axis == "x":
            self.rotate_matrix = [[1, 0, 0, 0],
                                  [0, cos(angel), sin(angel), 0],
                                  [0, -1 * sin(angel), cos(angel), 0],
                                  [0, 0, 0, 1]]
            color = "red"
        elif axis == "y":
            self.rotate_matrix = [[cos(angel), 0, -1 * sin(angel), 0],
                                  [0, 1, 0, 0],
                                  [sin(angel), 0, cos(angel), 0],
                                  [0, 0, 0, 1]]
            color = "yellow"
        else:
            raise Exception("Такой оси не существует!")

        new_points = self.multiply()
        new_cube = Cube(new_points,color)
        new_cube.remove1()
        self.remove1()
        return new_cube
```

Метод перемножения матриц:

```
    def multiply(self):
        p_rows = len(self.points)
        p_columns = len(self.points[0])
        r_rows = len(self.rotate_matrix)
        r_columns = len(self.rotate_matrix[0])
        if(p_columns != r_rows):
            raise Exception("Чета матрицы сломались акта")
        new_points = [[0 for i in range(r_columns)] for j in range(p_rows)]
        for i in range(p_rows):
            for j in range(r_columns):
                for k in range(r_rows):
                    new_points[i][j] +=
self.points[i][k]*self.rotate_matrix[k][j]
        return new_points
```

Метод отрисовки кубов:

```
    def draw(self, *cubes):

        fig, ax = plt.subplots()
        plt.subplots_adjust(bottom=0.35)
        self.ax = fig.add_subplot(111, projection='3d')
        i = 0
        for cube in cubes:
            if (i == 0):
                curr = cube
                i += 1
            self.ax.add_collection3d(
                Poly3DCollection(cube.getFaces(), facecolors=cube.color,
linewidths=1, edgecolors=cube.color, alpha=.5))
            if cube.max_point > self.max_point:
                self.max_point = cube.max_point
        self.ax.set_xlim([-1 * (self.max_point + 3), self.max_point + 3])
        self.ax.set_ylim([-1 * (self.max_point + 3), self.max_point + 3])
        self.ax.set_zlim([-1 * (self.max_point + 3), self.max_point + 3])

        self.ax.set_xlabel('X')
        self.ax.set_ylabel('Y')
        self.ax.set_zlabel('Z')

        self.ax.quiver(0, 0, 0, self.max_point + 5, 0, 0, color='red',
arrow_length_ratio=0.1)  # Ось X (красная)
        self.ax.quiver(0, 0, 0, 0, self.max_point + 5, 0, color='yellow',
arrow_length_ratio=0.1)  # Ось Y (зелёная)
        self.ax.quiver(0, 0, 0, 0, 0, self.max_point + 5, color='green',
arrow_length_ratio=0.1)  # Ось Z (синяя)

        # Поле ввода для оси X
        axbox_x = plt.axes([0.2, 0.2, 0.1, 0.05])
        self.x_box = TextBox(axbox_x, "Угол поворота:")

        # Поле ввода для оси Y
        axbox_y = plt.axes([0.2, 0.1, 0.1, 0.05])
        self.y_box = TextBox(axbox_y, "Ось:")

        # Кнопка для обновления графика
        axbutton = plt.axes([0.45, 0.01, 0.15, 0.05])
        button = Button(axbutton, 'отрисовать')

        button.on_clicked(self.update_graph)

        plt.show()
```

ВЫВОД

В ходе лабораторной работы мы освоили работу с модулем matplotlib на языке программирования Python. Была написана программа для отрисовки куба в трехмерных координатах, с возможностью поворота куба по всем трем осям координат, как в положительном, так и в отрицательном положении. Были изучены принципы осуществления перемещения трехмерных тел в пространстве. Было реализовано создание матрицы преобразования для поворотов вокруг осей и перемножения этой матрицы с матрицей координат трехмерной фигуры. В ходе проведения работы, выяснилось, что координатные оси не отображаются в matplotlib привычным нам образом, в связи с чем было принято решение дополнительно отрисовывать координатные оси на графике, для повышения наглядности.