**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Компьютерная графика»

**Лабораторная работа № 2**

Тема: Каркасная визуализация выпуклого многогранника. Удаление невидимых линий.

Студент: Попова Наталья Сергеевна

Группа: 08-305

Преподаватель: Чернышов Л.Н.

Дата: 12.10.2022

Оценка:

Москва, 2022

1. **Постановка задачи**

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 №17** - Наклонная призма с основанием - правильный 6-угольник.

.

1. **Описание программы**

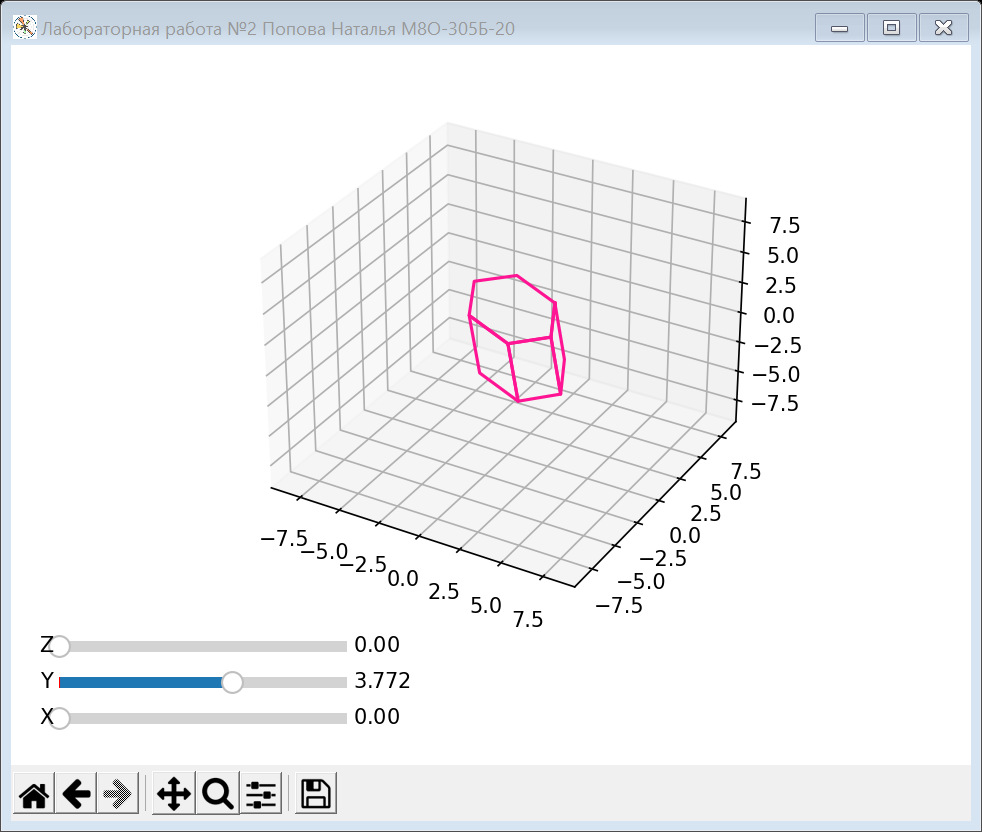
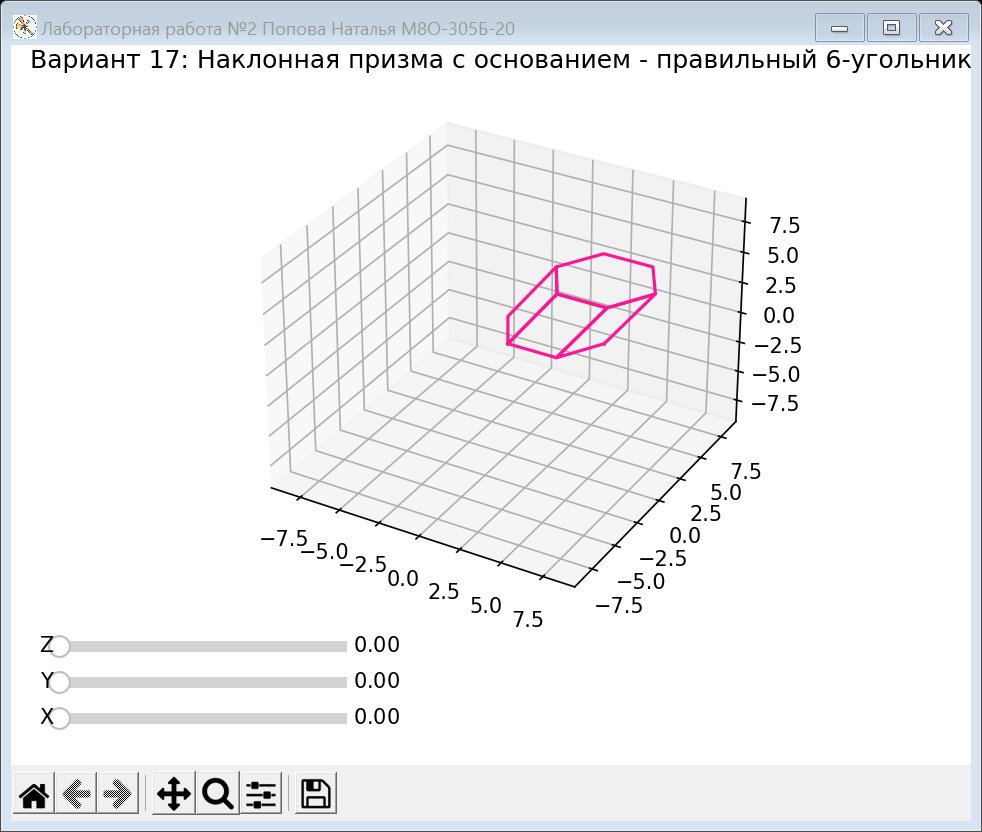
**Язык программирования**: Python

**Используемые библиотеки**: matplotlib

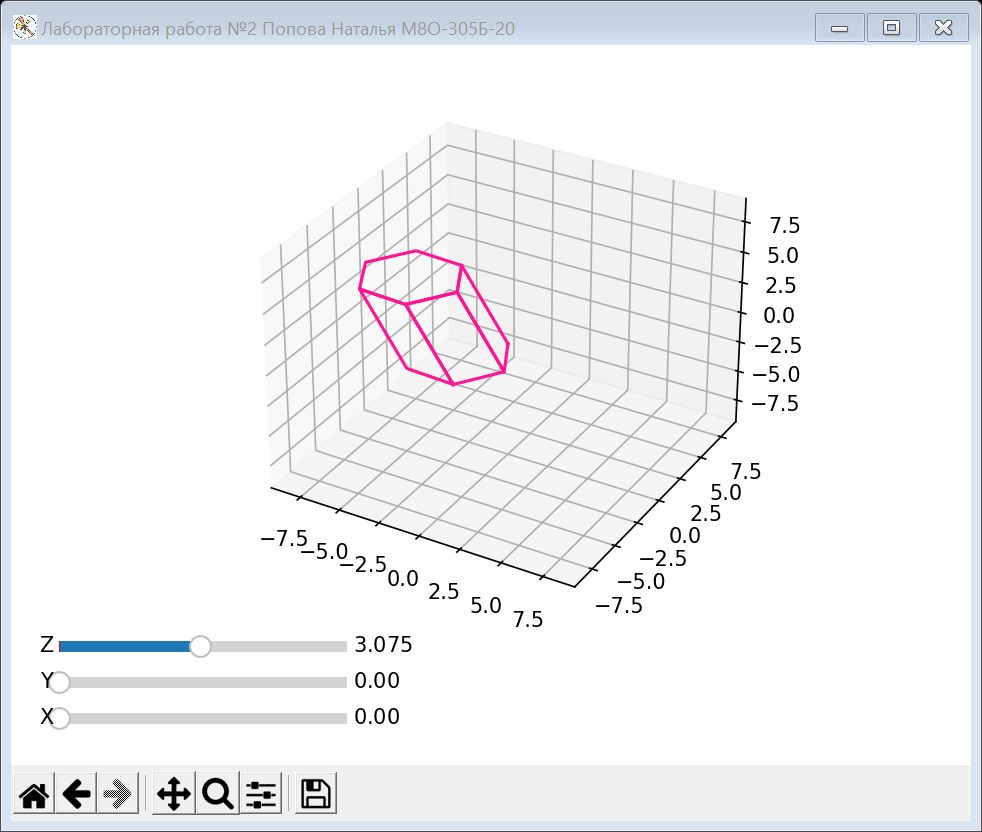
**Используемая среда программирования**: PyCharm

Данная программа рисует наклонную призму. В программе допускается возможность поворота фигуры вокруг трех осей (OX, OY, OZ). Также можно посмотреть проекции фигуры на плоскости OXY, OXZ, OYZ

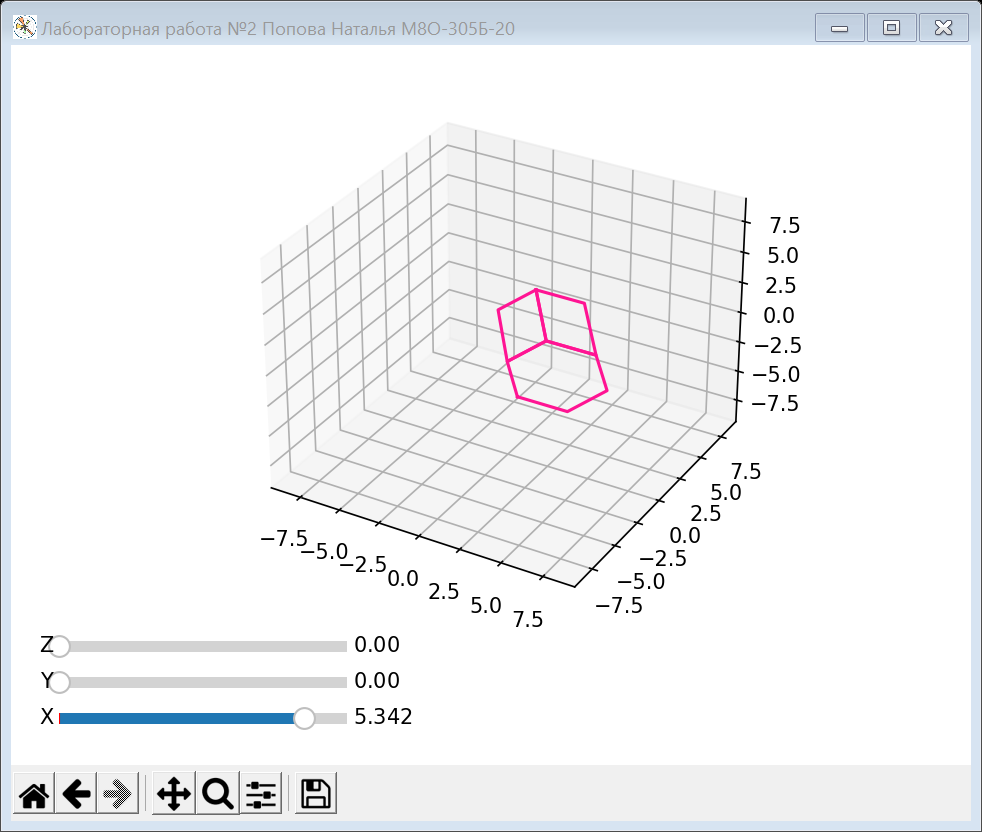
1. **Набор тестов**

****Тест №1 (угол поворота = 0)

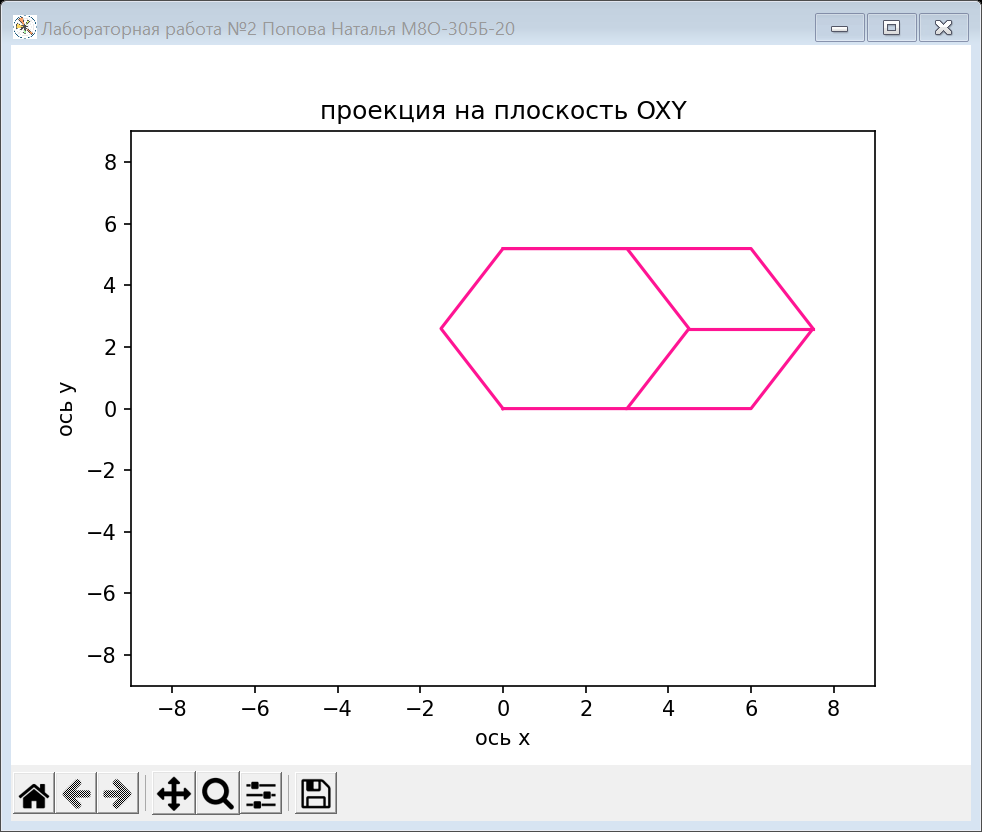
№2 (угол поворота вокруг оси OY = 3.772 рад)



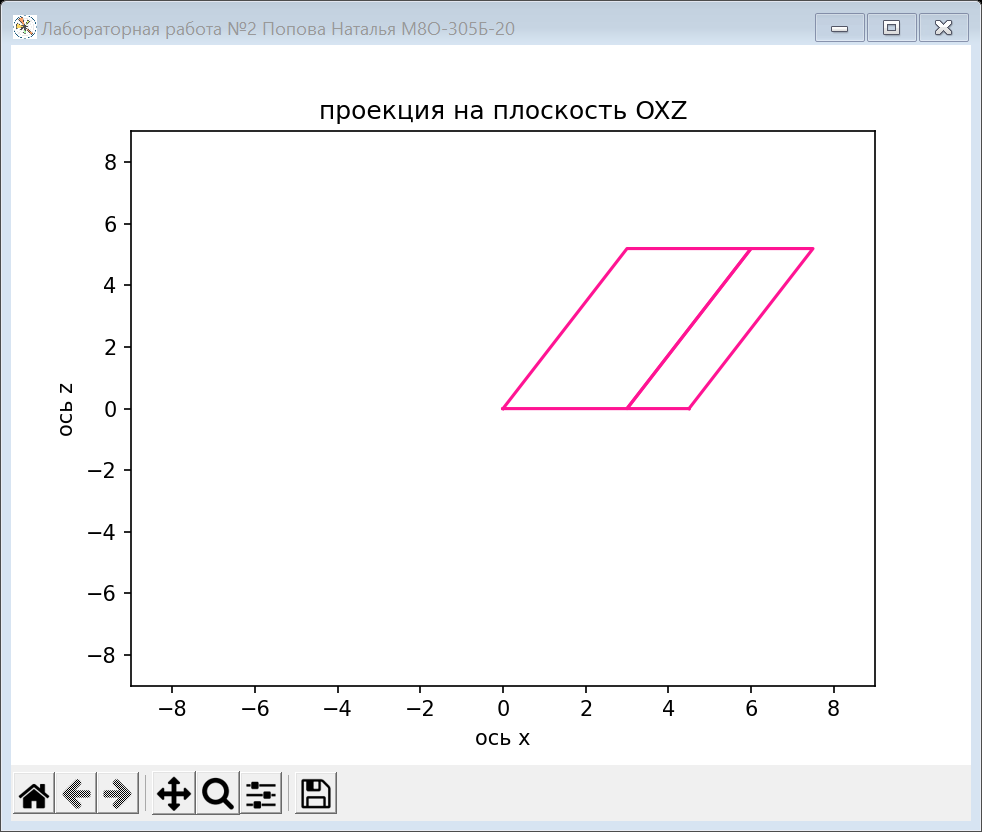
Тест №3 (угол поворота вокруг оси OZ = 3.075 рад)



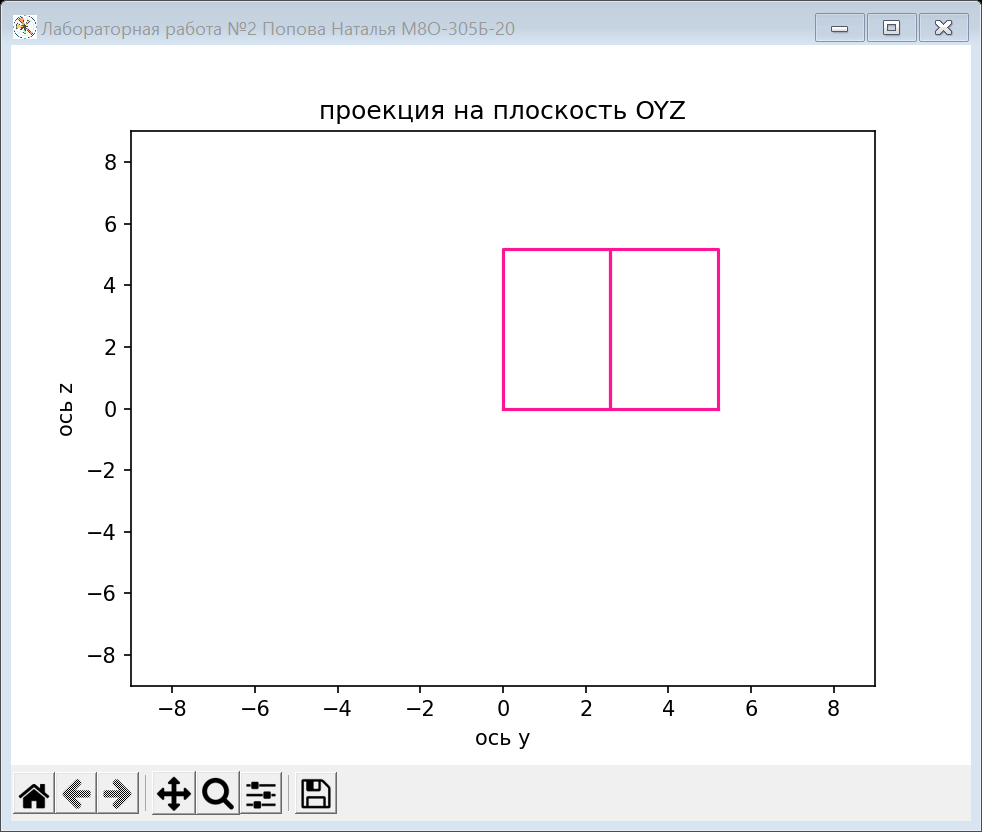
Тест №4 (угол поворота вокруг оси OX = 5.342 рад)



Тест №5 (проекция на плоскость OXY)



Тест №6 (проекция на плоскость OXZ)



Тест №7 (проекция на плоскость OYZ)

1. **Листинг программы**

*#Лабораторная работа №2 Попова Наталья М8О-305Б-20*import numpy as np  
import matplotlib.pyplot as plt  
import pylab  
from matplotlib.widgets import Slider, Button  
  
  
def get\_vertices(l,alpha, r):  
 A = np.empty((12, 3), dtype=float)  
 A [[0,1,2,3,4,5],2] = 0  
 A [[6,7,8,9,10,11],2] = l\*np.sin(alpha)  
 A [[0,1],1] = 0  
 A [[2,5],1] = np.sqrt(3)\*r/2  
 A[[3, 4], 1] = np.sqrt(3) \* r  
 A[[0,4],0] = 0  
 A[[1, 3], 0] = r  
 A[2, 0] = 3\*r/2  
 A[5,0] = -r/2  
 A[[6,7,8,9,10,11],1] = A [[0,1,2,3,4,5],1]  
 A[[6,7,8,9,10,11],0] = A [[0,1,2,3,4,5],0] + l\*np.cos(alpha)  
 return A  
  
F = np.array([0,1,2,3,4,5,6,7,8,9,10,11])  
K = [0,1,2]  
G = get\_vertices(6,np.pi/3,3)  
X = np.array (G[F,0])  
Y = np.array (G[F,1])  
Z = np.array (G[F,2])  
xx1 =0  
yy1=0  
zz1=0  
  
def check(v1x,v2x,v3x,v1y,v2y,v3y,v1z,v2z,v3z,w1,w2,w3):  
  
 Vec1x = v1x - v2x  
 Vec2x = v3x - v2x  
 Vec1y = v1y - v2y  
 Vec2y = v3y - v2y  
 Vec1z = v1z - v2z  
 Vec2z = v3z - v2z  
  
 A = Vec1y\*Vec2z-Vec2y\*Vec1z  
 B = Vec1z\*Vec2x-Vec2z\*Vec1x  
 C = Vec1x\*Vec2y-Vec2x\*Vec1y  
 D = -(A\*v1x + B\*v1y + C\*v1z)  
  
 m = -np.sign(A\*w1+B\*w2+C\*w3+D)  
  
 A = A\*m  
 B = B\*m  
 C = C\*m  
 D = D\*m  
  
 if (A\*7.5-B\*7.5+C\*7.5+D>0):  
 return 1  
 else:  
 return 0  
  
w1 = 0  
w2 = 0  
w3 = 0

1. for i in range (12):  
    w1 = w1+ G[i,0] / 12  
    w2 = w2 + G[i, 1] / 12  
    w3 = w3 + G[i, 2] / 12  
     
   W = np.array([w1,w2,w3])  
     
   l\_x = -9  
   r\_x = 9  
   l\_y = -9  
   r\_y = 9  
   l\_z = -9  
   r\_z = 9  
   fig = plt.figure('Лабораторная работа №2 Попова Наталья М8О-305Б-20')  
   ax = fig.add\_subplot(projection='3d')  
   ax.set\_title('Вариант 17: Наклонная призма с основанием - правильный 6-угольник')  
   ax.set\_xlim((l\_x, r\_x))  
   ax.set\_ylim((l\_y, r\_y))  
   ax.set\_zlim((l\_z, r\_z))  
   fig.subplots\_adjust(left=0.07, right=0.95, top=0.95, bottom=0.2)  
     
   axx1 = plt.axes([0.05, 0.05, 0.3, 0.03])  
   x1 = Slider(axx1, 'X', 0.0, 6.28, 0)  
   axy1 = plt.axes([0.05, 0.1, 0.3, 0.03])  
   y1 = Slider(axy1, 'Y', 0.0, 6.28, 0)  
   axz1 = plt.axes([0.05, 0.15, 0.3, 0.03])  
   z1 = Slider(axz1, 'Z', 0.0, 6.28, 0)  
     
   ax.plot(X[[0,1,7,6,0]],Y[[0,1,7,6,0]],Z[[0,1,7,6,0]], c='deeppink')  
   ax.plot(X[[2, 1, 7, 8, 2]], Y[[2, 1, 7, 8, 2]], Z[[2, 1, 7, 8, 2]], c='deeppink')  
   ax.plot(X[[0, 5, 11, 6, 0]], Y[[0, 5, 11, 6, 0]], Z[[0, 5, 11, 6, 0]], c='deeppink')  
   ax.plot(X[[6,7,8,9,10,11,6]], Y[[6,7,8,9,10,11,6]], Z[[6,7,8,9,10,11,6]], c='deeppink')  
     
   def plotXY():  
    l\_x = -9  
    r\_x = 9  
    l\_y = -9  
    r\_y = 9  
    fig1 = plt.figure('Лабораторная работа №2 Попова Наталья М8О-305Б-20')  
    ax = fig1.add\_subplot()  
    ax.set\_title('проекция на плоскость OXY')  
    ax.set\_xlim((l\_x, r\_x))  
    ax.set\_ylim((l\_y, r\_y))  
    plt.xlabel("ось x")  
    plt.ylabel("ось y")  
     
    F = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11])  
    G = get\_vertices(6, np.pi / 3, 3)  
    X = np.array(G[F, 0])  
    Y = np.array(G[F, 1])  
     
    ax.plot(X[[0, 5, 4, 3, 2, 1, 0]], Y[[0, 5, 4, 3, 2, 1, 0]], c='deeppink')  
    ax.plot(X[[1, 7, 8,9,10,4]], Y[[1, 7, 8,9,10,4]], c='deeppink')  
    ax.plot(X[[8,2]], Y[[8,2]], c='deeppink')  
     
    plt.show()  
     
   def plotXZ():  
    l\_x = -9  
    r\_x = 9  
    l\_y = -9  
    r\_y = 9  
    fig2 = plt.figure('Лабораторная работа №2 Попова Наталья М8О-305Б-20')  
    ax = fig2.add\_subplot()  
    ax.set\_title('проекция на плоскость OXZ')  
    ax.set\_xlim((l\_x, r\_x))  
    ax.set\_ylim((l\_y, r\_y))  
    plt.xlabel("ось x")  
    plt.ylabel("ось z")  
     
    ax.plot(X[[4, 3, 9, 10, 4]], Z[[4, 3, 9, 10, 4]], c='deeppink')  
    ax.plot(X[[2, 3, 9, 8, 2]], Z[[2, 3, 9, 8, 2]], c='deeppink')  
     
    plt.show()  
     
   def plotYZ():  
    l\_x = -9  
    r\_x = 9  
    l\_y = -9  
    r\_y = 9  
    fig3 = plt.figure('Лабораторная работа №2 Попова Наталья М8О-305Б-20')  
    ax = fig3.add\_subplot()  
    ax.set\_title('проекция на плоскость OYZ')  
    ax.set\_xlim((l\_x, r\_x))  
    ax.set\_ylim((l\_y, r\_y))  
    plt.xlabel("ось y")  
    plt.ylabel("ось z")  
     
    ax.plot(Y[[0, 5, 11, 6, 0]], Z[[0, 5, 11, 6, 0]], c='deeppink')  
    ax.plot(Y[[4, 5, 11, 10, 4]], Z[[4, 5, 11, 10, 4]], c='deeppink')  
     
    plt.show()  
      
   def update():  
     
    yy1 = y1.val  
    zz1 = z1.val  
    xx1 = x1.val  
    F = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11])  
    G = get\_vertices(6, np.pi / 3, 3)  
    X = np.array(G[F, 0])  
    Y = np.array(G[F, 1])  
    Z = np.array(G[F, 2])  
     
    if (xx1 != 0):  
    G[F,1] = G[F,1] \* np.cos(xx1) + G[F,2] \* np.sin(xx1)  
    G[F, 2] = -G[F, 1] \* np.sin(xx1) + G[F, 2] \* np.cos(xx1)  
     
    elif (yy1 != 0):  
    G[F,0] = G[F,0] \* np.cos(yy1) - G[F,2] \* np.sin(yy1)  
    G[F,2] = G[F, 0] \* np.sin(yy1) + G[F, 2] \* np.cos(yy1)  
     
    elif (zz1 != 0):  
    G[F, 0] = G[F,0] \* np.cos(zz1) - G[F,1] \* np.sin(zz1)  
    G[F, 1] = -G[F, 0] \* np.sin(zz1) + G[F, 1] \* np.cos(zz1)  
     
    w1 = 0  
    w2 = 0  
    w3 = 0  
     
    for i in range(12):  
    w1 = w1 + G[i, 0] / 12  
    w2 = w2 + G[i, 1] / 12  
    w3 = w3 + G[i, 2] / 12  
    X = np.array(G[F, 0])  
    Y = np.array(G[F, 1])  
    Z = np.array(G[F, 2])  
    ax.clear()  
    l\_x = -9  
    r\_x = 9  
    l\_y = -9  
    r\_y = 9  
    l\_z = -9  
    r\_z = 9  
     
    ax.set\_xlim((l\_x, r\_x))  
    ax.set\_ylim((l\_y, r\_y))  
    ax.set\_zlim((l\_z, r\_z))  
    fig.subplots\_adjust(left=0.07, right=0.95, top=0.95, bottom=0.2)  
     
    a = check(X[0],X[1],X[7],Y[0],Y[1],Y[7],Z[0],Z[1],Z[7],w1,w2,w3)  
    if (a == 1):  
    ax.plot(X[[0,1,7,6,0]],Y[[0,1,7,6,0]],Z[[0,1,7,6,0]], c='deeppink')  
     
    a = check(X[2], X[1], X[7], Y[2], Y[1], Y[7], Z[2], Z[1], Z[7], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[2, 1, 7, 8, 2]], Y[[2, 1, 7, 8, 2]], Z[[2, 1, 7, 8, 2]], c='deeppink')  
     
    a = check(X[2], X[3], X[9], Y[2], Y[3], Y[9], Z[2], Z[3], Z[9], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[2, 3, 9, 8, 2]], Y[[2, 3, 9, 8, 2]], Z[[2, 3, 9, 8, 2]], c='deeppink')  
     
    a = check(X[4], X[3], X[9], Y[4], Y[3], Y[9], Z[4], Z[3], Z[9], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[4, 3, 9, 10, 4]], Y[[4, 3, 9, 10, 4]], Z[[4, 3, 9, 10, 4]], c='deeppink')  
     
    a = check(X[4], X[5], X[11], Y[4], Y[5], Y[11], Z[4], Z[5], Z[11], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[4, 5, 11, 10, 4]], Y[[4, 5, 11, 10, 4]], Z[[4, 5, 11, 10, 4]], c='deeppink')  
     
    a = check(X[0], X[5], X[11], Y[0], Y[5], Y[11], Z[0], Z[5], Z[11], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[0, 5, 11, 6, 0]], Y[[0, 5, 11, 6, 0]], Z[[0, 5, 11, 6, 0]], c='deeppink')  
     
    a = check(X[0], X[5], X[4], Y[0], Y[5], Y[4], Z[0], Z[5], Z[4], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[0, 5, 4, 3, 2,1,0]], Y[[0, 5, 4, 3, 2,1,0]], Z[[0, 5, 4, 3, 2,1,0]], c='deeppink')  
    a = check(X[6], X[7], X[8], Y[6], Y[7], Y[8], Z[6], Z[7], Z[8], w1, w2, w3)  
    if (a == 1):  
    ax.plot(X[[6,7,8,9,10,11,6]], Y[[6,7,8,9,10,11,6]], Z[[6,7,8,9,10,11,6]], c='deeppink')  
     
    ax.grid()  
    plt.show()  
     
   def on\_change(val):  
    update()  
     
   x1.on\_changed(on\_change)  
   y1.on\_changed(on\_change)  
   z1.on\_changed(on\_change)  
     
   plt.show()

ЛИТЕРАТУРА

1. Справочник по использованию 3D графики на Python [электронный ресурс].URL: <https://matplotlib.org/stable/tutorials/toolkits/mplot3d.html>
2. 3D моделирование в Python [электронный ресурс].

URL: https://habr.com/ru/post/572760/?ysclid=l906vcgm6l871141223

1. Электронный учебник по компьютерной графике[электронный ресурс].

URL: <https://ychebnikkompgrafblog.wordpress.com/>