**Минобрнауки РФ**

**ЮЗГУ**

**ФФиПИ**

Кафедра программной инженерии

**ЛАБОРАТОРНАЯ РАБОТА №5**

**ПРОЕКТИВНЫЕ ПРЕОБРАЗОВАНИЯ**

Выполнил: ст. гр. ПО-22б

Алешин К.Р.

Проверил: Ефремов В. В.

Курск, 2024 г

**Задание:**  
****

**Листинг программы:**import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d.art3d import Line3DCollection

def create\_cube():

"""Создаёт вершины и рёбра каркасного куба."""

vertices = np.array([

[-1, -1, -1],

[-1, -1, 1],

[-1, 1, -1],

[-1, 1, 1],

[ 1, -1, -1],

[ 1, -1, 1],

[ 1, 1, -1],

[ 1, 1, 1],

])

edges = [

(0, 1), (0, 2), (0, 4), # Из вершины 0

(1, 3), (1, 5), # Из вершины 1

(2, 3), (2, 6), # Из вершины 2

(3, 7), # Из вершины 3

(4, 5), (4, 6), # Из вершины 4

(5, 7), # Из вершины 5

(6, 7) # Из вершины 6

]

return vertices, edges

def create\_dodecahedron():

"""Создаёт вершины и рёбра каркасного додекаэдра."""

phi = (1 + np.sqrt(5)) / 2 # Золотое сечение

# Вершины додекаэдра

vertices = np.array([

[-1, -1, -1], [1, -1, -1], [-1, 1, -1], [1, 1, -1], # Нижняя и верхняя основания

[-1, -1, 1], [1, -1, 1], [-1, 1, 1], [1, 1, 1], # Нижняя и верхняя основания

[0, -1/phi, -phi], [0, -1/phi, phi], [0, 1/phi, -phi], [0, 1/phi, phi], # Углы

[-1/phi, -phi, 0], [1/phi, -phi, 0], [-1/phi, phi, 0], [1/phi, phi, 0],

[-phi, 0, -1/phi], [phi, 0, -1/phi], [-phi, 0, 1/phi], [phi, 0, 1/phi]

])

# Рёбра додекаэдра (индексы вершин)

edges = [

(0, 8), (0, 12), (0, 16), # Из вершины 0

(1, 8), (1, 13), (1, 17), # Из вершины 1

(2, 10), (2, 14), (2, 16), # Из вершины 2

(3, 10), (3, 15), (3, 17), # Из вершины 3

(4, 9), (4, 12), (4, 18), # Из вершины 4

(5, 9), (5, 13), (5, 19), # Из вершины 5

(6, 11), (6, 14), (6, 18), # Из вершины 6

(7, 11), (7, 15), (7, 19), # Из вершины 7

(8, 10), (9, 11), (12, 13), (14, 15), (16, 18), (17, 19) # Рёбра между соседними вершинами

]

return vertices, edges

def plot\_polyhedron(vertices, edges, ax, color="blue"):

"""Рисует каркасное изображение многогранника и подписывает вершины."""

lines = [(vertices[start], vertices[end]) for start, end in edges]

line\_collection = Line3DCollection(lines, colors=color, linewidths=1)

ax.add\_collection3d(line\_collection)

# Подписываем вершины

for i, vertex in enumerate(vertices):

ax.text(vertex[0], vertex[1], vertex[2], f"{i}", color="red", fontsize=10, ha='right')

ax.scatter(vertices[:, 0], vertices[:, 1], vertices[:, 2], color="red", s=10)

def main():

# Создаём фигуру

fig = plt.figure(figsize=(10, 5))

# Добавляем график для куба

ax\_cube = fig.add\_subplot(121, projection="3d")

vertices\_cube, edges\_cube = create\_cube()

plot\_polyhedron(vertices\_cube, edges\_cube, ax\_cube, color="blue")

ax\_cube.set\_title("Куб")

ax\_cube.set\_box\_aspect([1, 1, 1])

# Добавляем график для додекаэдра

ax\_dodecahedron = fig.add\_subplot(122, projection="3d")

vertices\_dodecahedron, edges\_dodecahedron = create\_dodecahedron()

plot\_polyhedron(vertices\_dodecahedron, edges\_dodecahedron, ax\_dodecahedron, color="green")

ax\_dodecahedron.set\_title("Додекаэдр")

ax\_dodecahedron.set\_box\_aspect([1, 1, 1])

# Установка пределов для осей

for ax in [ax\_cube, ax\_dodecahedron]:

ax.set\_xlim([-2, 2])

ax.set\_ylim([-2, 2])

ax.set\_zlim([-2, 2])

ax.set\_xlabel("X")

ax.set\_ylabel("Y")

ax.set\_zlabel("Z")

# Отображаем графики

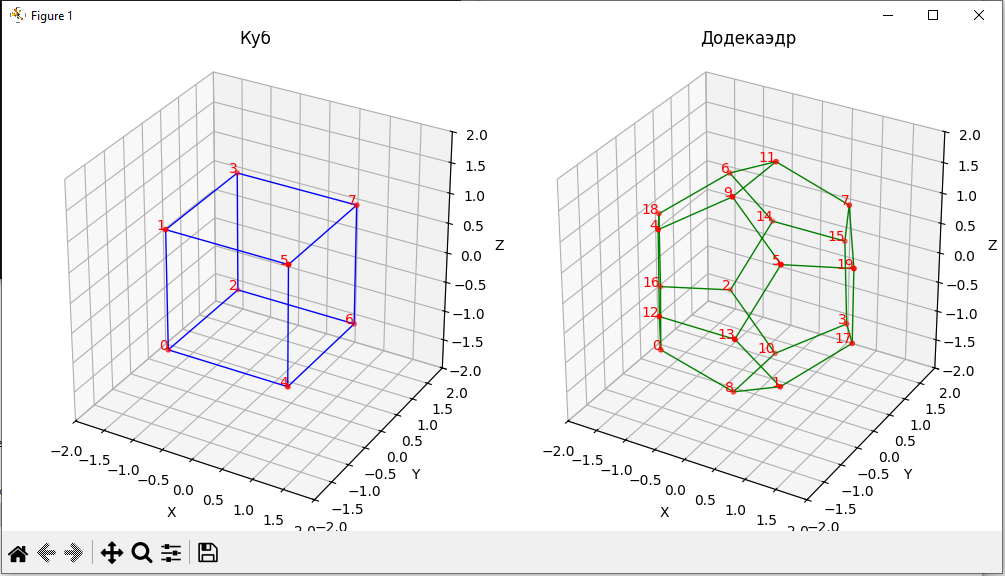
plt.tight\_layout()

plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Пример работы программы:**

****