Отчет по домашнему заданию по дисциплине ПиКЯП
«3D визуализация объектов с помощью инструмента OpenGL»

Студент: Абдуллаев Г. А.

Преподаватель: Нардид А. Н.

Main.py

```
import pygame
import sys
from pygame.locals import *
from OpenGL.GL import *
from OpenGL.GLU import *

vertextxt = "vertex.txt"
facestxt = "faces.txt"

paints = [
    (0, 255, 0),    # green
    (255, 0, 0),    # red
    (255, 255, 0),  # yellow
    (0, 255, 255),  # cyan
    (0, 0, 255),    # blue
    (255, 255, 255) # white
]

def get_list(txtname):
    listname = []
    with open(txtname) as f:
        for line in f:
            line = line.rstrip(",\r\n").replace("(", ").replace(")", "").replace(" ", ")
            row = list(line.split(","))
            listname.append(row)
    listname = [[float(j) for j in i] for i in listname]
    return listname
```

```
modelVerts = get_list(vertextxt)
modelFaces = get_list(facestxt)


#GL_LINES Обрабатывает каждую пару вершин как независимый сегмент линии.
#Вершины 2n - 1 и 2n определяют строку n. Рисуются N/2 линии.
#Функция glClear очищает буферы до предустановленных значений.


#GL_TRIANGLES Рассматривает каждый триплет вершин как независимый треугольник.
#ершины 3n - 2, 3n - 1 и 3n определяют треугольник n.
#Рисуются N/3 треугольников.


#GL_COLOR_BUFFER_BIT - Буферы в настоящее время включены для записи цветов.
#GL_DEPTH_BUFFER_BIT - Буфер глубины.


#https://learn.microsoft.com/ru-ru/windows/win32/opengl/glenable
#https://learn.microsoft.com/ru-ru/windows/win32/opengl/glbegin



def drawfaces(curColorIndx):
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glBegin(GL_TRIANGLES)
    for eachface in modelFaces:
        for eachvert in eachface:
            glColor3fv(paints[curColorIndx])
            glVertex3fv(modelVerts[int(eachvert)])
    glEnd()
```

```
def main():
    pygame.init()
    display = (800, 800)
    pygame.display.set_caption("RENDERING OBJECT")
    FPS = pygame.time.Clock()
    pygame.display.set_mode(display, DOUBLEBUF | OPENGL)
    gluPerspective(45, 1, 0.1, 50)
    glTranslate(0, 0, -5)
    glRotate(-90, 1, 0, 0)

    Left = False
    Right = False
    Up = False
    Down = False
    curColorIndx = 0
    def moveOBJ():
        if Left:
            glRotate(-1, 0, 0, 1)
        if Right:
            glRotate(1, 0, 0, 1)
        if Up:
            glRotate(1, 1, 0, 0)
        if Down:
            glRotate(-1, 1, 0, 0)

    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
```

```
    pygame.quit()
    sys.exit()
if event.type == KEYDOWN:
    if event.key == K_ESCAPE:
        pygame.quit()
        sys.exit()
    if event.key == K_a:
        Left = True
    if event.key == K_d:
        Right = True
    if event.key == K_w:
        Up = True
    if event.key == K_s:
        Down = True
    if event.key == K_e:
        if (curColorIndx < 5):
            curColorIndx = curColorIndx + 1
        elif (curColorIndx == 5):
            curColorIndx = 0
if event.type == KEYUP:
    if event.key == K_a:
        Left = False
    if event.key == K_d:
        Right = False
    if event.key == K_w:
        Up = False
    if event.key == K_s:
        Down = False
        pygame.display.flip()
        drawfaces(curColorIndx)
        moveOBJ()
        FPS.tick(60)

main()
```

Develop.py

```
import pygame
import sys
from pygame.locals import *
from OpenGL.GL import *
from OpenGL.GLU import *

vertextxt = "vertex.txt"
facestxt = "faces.txt"

paints = [
    (0, 255, 0),    # green
    (255, 0, 0),    # red
    (255, 255, 0),  # yellow
    (0, 255, 255),  # cyan
    (0, 0, 255),    # blue
    (255, 255, 255) # white
]

def get_list(txtname):
    listname = []
    with open(txtname) as f:
        for line in f:
            line = line.rstrip(",\r\n").replace("(", ").replace(")", "").replace(" ", ")
            row = list(line.split(","))
            listname.append(row)
    listname = [[float(j) for j in i] for i in listname]
    return listname

modelVerts = get_list(vertextxt)
modelFaces = get_list(facestxt)

def draw_faces(current_color):
    glEnable(GL_POLYGON_OFFSET_FILL)
    glPolygonOffset(1.0, 1.0)
    glBegin(GL_TRIANGLES)
    for eachface in modelFaces:
        for eachvert in eachface:
            glColor3fv(current_color)
            glVertex3fv(modelVerts[int(eachvert)])
    glEnd()
    glDisable(GL_POLYGON_OFFSET_FILL)

def draw_edges():
    glColor3f(0, 0, 0)
    glPolygonMode(GL_FRONT_AND_BACK, GL_LINE)
    glBegin(GL_TRIANGLES)
    for eachface in modelFaces:
        for eachvert in eachface:
            glVertex3fv(modelVerts[int(eachvert)])
    glEnd()
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL)

def main():
    pygame.init()
    display = (800, 800)
    pygame.display.set_caption("edgesejji")
    FPS = pygame.time.Clock()
    pygame.display.set_mode(display, DOUBLEBUF | OPENGL)
    gluPerspective(60, 1, 0.1, 50)
    glTranslate(0, 0, -5)
    glRotate(-90, 1, 0, 0)

    glEnable(GL_DEPTH_TEST)
    glEnable(GL_CULL_FACE)
    glCullFace(GL_BACK)

    Left = False
    Right = False
    Up = False
    Down = False
    color_index = 0

    def moveOBJ():
        if Left:
            glRotate(-1, 0, 0, 1)
        if Right:
            glRotate(1, 0, 0, 1)
        if Up:
```

```
        glRotate(1, 1, 0, 0)
    if Down:
        glRotate(-1, 1, 0, 0)

current_color = paints[color_index]

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
        if event.type == KEYDOWN:
            if event.key == K_ESCAPE:
                pygame.quit()
                sys.exit()
            if event.key == K_a:
                Left = True
            if event.key == K_d:
                Right = True
            if event.key == K_w:
                Up = True
            if event.key == K_s:
                Down = True
            if event.key == K_c:
                color_index = (color_index + 1) % len(paints)
                current_color = paints[color_index]
        if event.type == KEYUP:
            if event.key == K_a:
                Left = False
                if event.key == K_d:
                    Right = False
                if event.key == K_w:
                    Up = False
                if event.key == K_s:
                    Down = False

        pygame.display.flip()
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
        draw_faces(current_color)
        draw_edges()
        moveOBJ()
        FPS.tick(60)

main()
```

Light.py

```
import pygame
import sys
from pygame.locals import *
from OpenGL.GL import *
from OpenGL.GLU import *

vertextxt = "vertex.txt"
facestxt = "faces.txt"

paints = [
    (0, 255, 0),    # green
    (255, 0, 0),    # red
    (255, 255, 0),  # yellow
    (0, 255, 255),  # cyan
    (0, 0, 255),    # blue
    (255, 255, 255) # white
]

def get_list(txtname):
    listname = []
    with open(txtname) as f:
        for line in f:
            line = line.rstrip(",\r\n").replace("(", ").replace(")", "").replace(" ", ")
            row = list(line.split(","))
            listname.append(row)
    listname = [[float(j) for j in i] for i in listname]
    return listname

modelVerts = get_list(vertextxt)
modelFaces = get_list(facestxt)

def draw_faces(current_color):
    glEnable(GL_POLYGON_OFFSET_FILL)
    glPolygonOffset(1.0, 1.0)
    glBegin(GL_TRIANGLES)
    for eachface in modelFaces:
        for eachvert in eachface:
            glVertex3fv(modelVerts[int(eachvert)])
    glEnd()
    glDisable(GL_POLYGON_OFFSET_FILL)
```

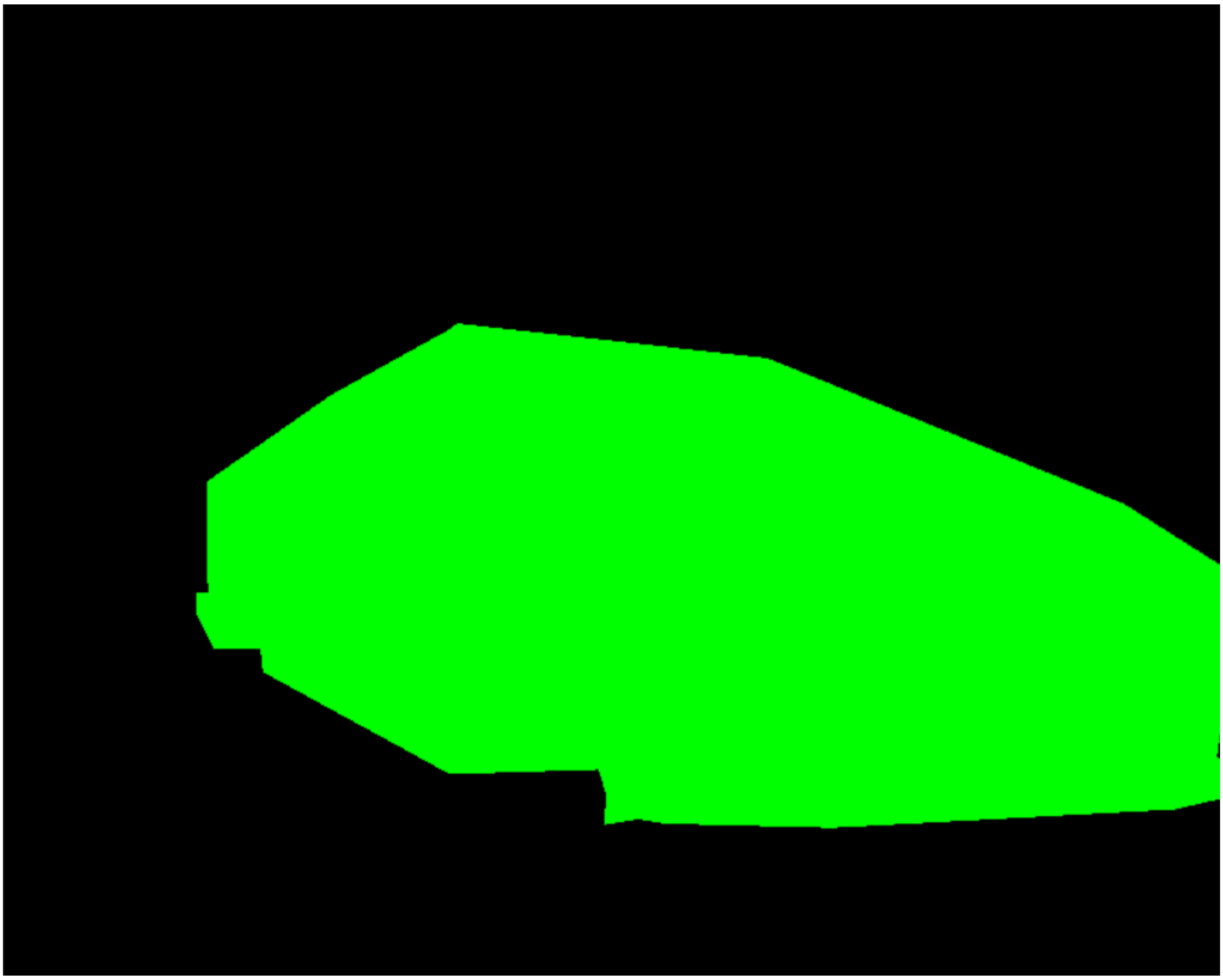
```
def draw_edges():
    glColor3f(0, 0, 0)
    glPolygonMode(GL_FRONT_AND_BACK, GL_LINE)
    glBegin(GL_TRIANGLES)
    for eachface in modelFaces:
        for eachvert in eachface:
            glVertex3fv(modelVerts[int(eachvert)])
    glEnd()
    glPolygonMode(GL_FRONT_AND_BACK, GL_FILL)


def setup_lighting():
    glEnable(GL_LIGHTING)
    glEnable(GL_LIGHT0)
    glLightfv(GL_LIGHT0, GL_POSITION, [32, 32, 32, 32])
    glLightfv(GL_LIGHT0, GL_AMBIENT, [0.25, 0.25, 0.25, 1])
    glLightfv(GL_LIGHT0, GL_DIFFUSE, [0.8, 0.8, 0.8, 1])
    glLightfv(GL_LIGHT0, GL_SPECULAR, [32, 32, 32, 32])

    # material
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE, [1,
1, 1, 1])
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, [1, 1, 1, 1])
    glMaterialf(GL_FRONT_AND_BACK, GL_SHININESS, 50)


def main():
    pygame.init()
    display = (900, 850)
    pygame.display.set_caption("LIGHT")
```

```
FPS = pygame.time.Clock()
pygame.display.set_mode(display, DOUBLEBUF | OPENGL)
gluPerspective(45, 1, 0.1, 50)
glTranslate(0, 0, -5)
glRotate(-90, 1, 0, 0)

glEnable(GL_DEPTH_TEST)
glEnable(GL_CULL_FACE)
glCullFace(GL_BACK)

setup_lighting()

Left = False
Right = False
Up = False
Down = False
color_index = 0

def moveOBJ():
    if Left:
        glRotate(-1, 0, 0, 1)
    if Right:
        glRotate(1, 0, 0, 1)
    if Up:
        glRotate(1, 1, 0, 0)
    if Down:
        glRotate(-1, 1, 0, 0)

current_color = paints[color_index]
```

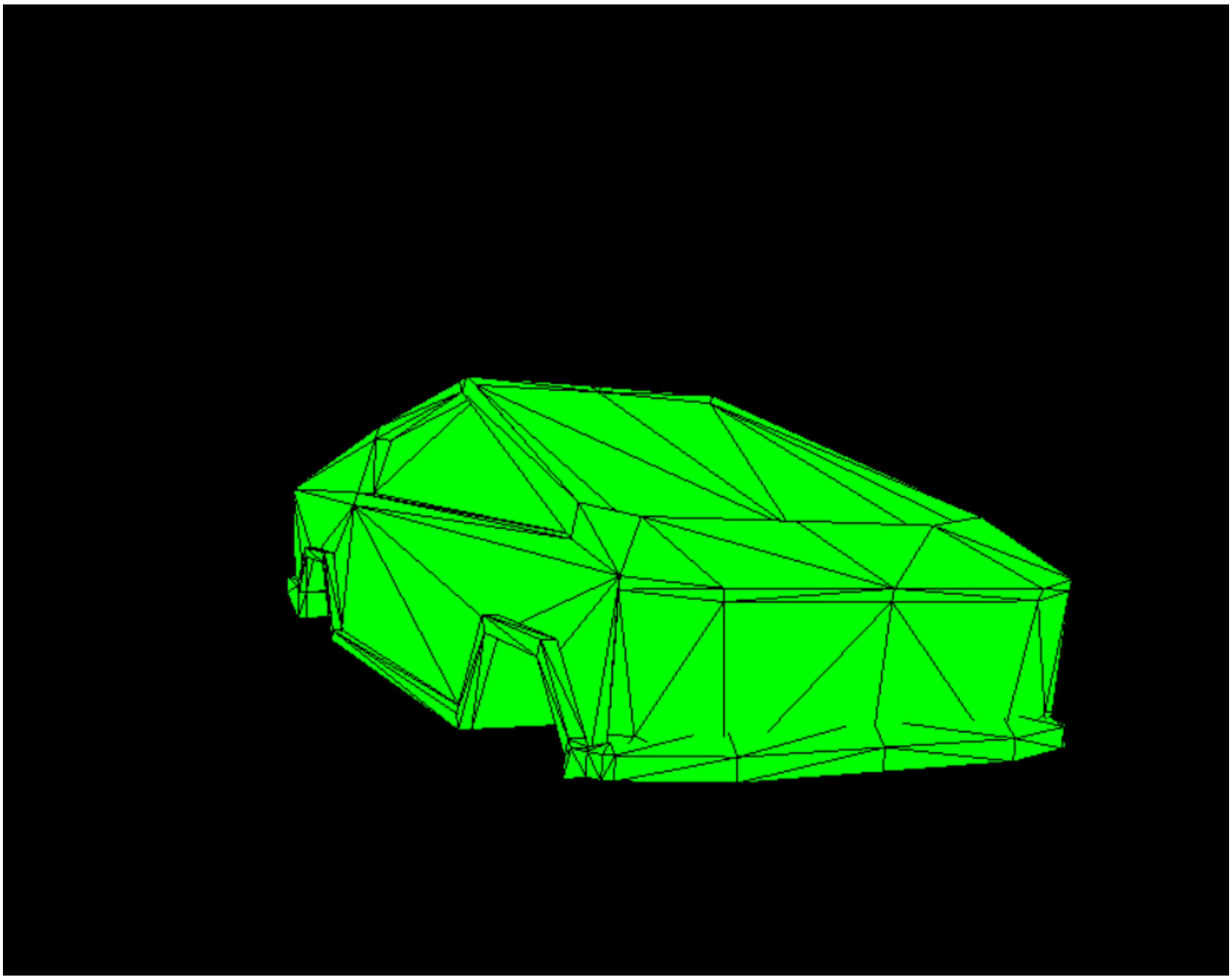
```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            sys.exit()
        if event.type == KEYDOWN:
            if event.key == K_ESCAPE:
                pygame.quit()
                sys.exit()
            if event.key == K_a:
                Left = True
            if event.key == K_d:
                Right = True
            if event.key == K_w:
                Up = True
            if event.key == K_s:
                Down = True
            if event.key == K_c:
                color_index = (color_index + 1) % len(paints)
                current_color = paints[color_index]
        if event.type == KEYUP:
            if event.key == K_a:
                Left = False
            if event.key == K_d:
                Right = False
            if event.key == K_w:
                Up = False
            if event.key == K_s:
```

```
            Down = False

    pygame.display.flip()
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    draw_faces(current_color)
    draw_edges()
    moveOBJ()
    FPS.tick(60)

main()
```

Screenshots:
main.py



Developing.py



Light.py: