Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

LAB-1

[Code No.: COMP 342]

Submitted by

Sudip Bhattarai

Roll no:11

Submitted to

Mr. Dhiraj Shrestha

A 18, 2024

- **Mention the name of Programming language and Graphics Library you are using this semester for performing your Computer Graphics Lab and Project**

We will be using python for programming language and OpenGL as graphics library for both project and lab.

- **Write the code snippets for setting graphics environment in your chosen graphics library and display the resolution of your display system through functions/classes provided by your graphics library**

Here I have simply installed the dependencies of PyOpenGL and GLFW which are used with PYOpenGL to manage windows and create open gl contexts.

After installation of all the above mentioned packages I have simply imported them to use in my code.

Command to install PYOpenGL:-

pip3 install PyOpenGL

Importing it to use in my code:-

*from OpenGL.GL import **

For displaying resolution as already mention i have used glfw

Command to install glfw:

pip install glfw

Importing it to use in my code:-

import glfw

For displaying resolution we have used the function defined in glfw to get necessary screen width and height of my device the code snippet is shown below

```
import glfw

def showResoultion():
    if not glfw.init():
        return
    monitor=glfw.get_primary_monitor()
    vidmode =glfw.get_video_mode(monitor)
    print('Resoultion',vidmode.size.width,'*',vidmode.size.height)

if __name__=='__main__':
    showResoultion()
```

The output can be seen as:

```
(.venv) sudipbhattarai@Sudips-MacBook-Pro Graphics Lab % python resolution.py
Resoultion 1680 * 1050
```

- **Get Familiar with the coordinate system and Draw a Logo of Kathmandu University given below using the chosen Graphics geometric functions/ classes provided by the your chosen graphics library and also color the Logo accordingly**

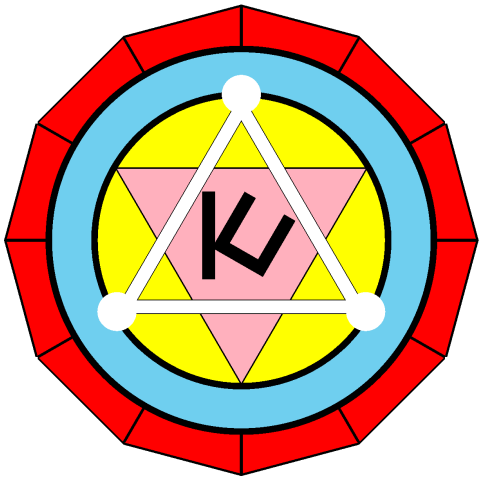
Here to be familiar with coordinate system I have created window of 1000*1000 using glfw create window function.also here the below line

```
glOrtho(0, x_center*2 ,y_center*2, 0, 0, 1)
```

creates a projection matrix, which maps 0 to the left border of the viewport, x center*2 to the right, y_center*2 to the bottom and 0 to the top of the viewport.

Now here we have divided the circle into different shapes and functions to achieve the full logo of KU show. Let me show you with an output and that code snippet that is drawing each component of the logo.

Output:



Here now lets begin with the outer red part which is drawn by drawing a line for generating black line and drawing the polygon to fill the red color with a total of 12 segments .

```
def red_polygon(x_coordinate, y_coordinate, radius, segments):
    #for  drawing black line outside the polygon
    glLineWidth(12.0)
    glBegin(GL_LINES)
    glColor4fv(black)
    delta_theta = 2.0 * pi / segments
    for i in range(segments):
        theta1 = delta_theta * i
        theta2 = delta_theta * (i + 1)
        x1 = radius * cos(theta1) + x_coordinate
        y1 = radius * sin(theta1) + y_coordinate
        x2 = radius * cos(theta2) + x_coordinate
        y2 = radius * sin(theta2) + y_coordinate

        glVertex2f(x1, y1)
        glVertex2f(x2, y2)
    glEnd()

    #for filling the polygon with red color
    glBegin(GL_POLYGON)
    glColor4fv(red)
    delta_theta = 2.0 * pi / segments
    for i in range(segments):
        theta1 = delta_theta * i
        theta2 = delta_theta * (i + 1)
        x1 = radius * cos(theta1) + x_coordinate
        y1 = radius * sin(theta1) + y_coordinate
        x2 = radius * cos(theta2) + x_coordinate
        y2 = radius * sin(theta2) + y_coordinate

        glVertex2f(x1, y1)
        glVertex2f(x2, y2)
    glEnd()
```

Now we have divide the read polygon into 12 segments and drawn perpendicular line which coordinate calculation is show below

```
#divider line
def divider_lines(x_coordinate, y_coordinate, radius, num_segments):
    glLineWidth(8.0)
    glBegin(GL_LINES)
    glColor4fv(black)
    delta_theta = 2.0 * pi / num_segments
    for i in range(num_segments):
        theta = delta_theta * i
        x1 = x_coordinate
        y1 = y_coordinate
        x2 = radius* cos(theta) + x_coordinate
        y2 = radius * sin(theta) + y_coordinate
        glVertex2f(x1, y1)
        glVertex2f(x2, y2)
    glEnd()
```

Now we have drawn a circle with a different color and radius which is achieved using the function below.

```
#draws circle
def draw_circle(x_cordinate, y_cordinate, radius, segment, color):
    glBegin(GL_POLYGON)
    glColor4fv(color)
    for i in range(segment):
        theta = 2.0 * pi * i / segment
        x = radius * cos(theta) + x_cordinate
        y = radius * sin(theta) + y_cordinate
        glVertex2f(x, y)
    glEnd()
```

Finally for center we have defined two functions to draw upside down and normal triangle with hollow shape at middle.In both I have used line to create a outline and polygon to fill in the colors with.

Now for drawing the central KU we have drawn a rectangle and U shape as similar to triangle above i.e by using polygon also we have rotated U shape by 30 degrees .
Here let me explain 3 lines of code

- glTranslateF(X,Y,0): This translates the coordinate system to (X,Y)
- glRotateF(rotation,X,Y,Z): This rotates the coordinate system by rotation angle
- glTranslatef(-X,-Y,0): retranslates the coordinate system to the original coordinate system .

```
def draw_rectangle(x, y, width, height):
    glColor4fv(black)
    glBegin(GL_QUADS)
    glVertex2f(x, y)
    glVertex2f(x + width, y)
    glVertex2f(x + width, y + height)
    glVertex2f(x, y + height)
    glEnd()



def draw_u_shape(x, y, width, height, space, rotation):
    pivot_x = x
    pivot_y = y
    glTranslatef(pivot_x, pivot_y, 0)
    glRotatef(rotation, 0, 0, 1)
    glTranslatef(-pivot_x, -pivot_y, 0)

    glColor4fv(black)

    glBegin(GL_QUADS)
    glVertex2f(x, y)
    glVertex2f(x, y + height + 20)
    glVertex2f(x+width,y + height + 20)
    glVertex2f(x+width, y)
    glEnd()

    glBegin(GL_QUADS)
    glVertex2f(x+width,y+height)
    glVertex(x+width+space,y+height)
    glVertex(x+width+space,y+height+20)
    glVertex2f(x+width,y+height+20)
    glEnd()

    glBegin(GL_QUADS)
    glVertex(x+width+space,y+height+20)
    glVertex(x+2*width+space,y+height+20)
    glVertex(x+2*width+space,y)
    glVertex(x+width+space,y)
    glEnd()
```

All these functions are called in sequence to achieve the above output which is shown above.
And the sequence of function call to achieve this is shown below:

```
def main():
    if not glfw.init():
        print("Failed to initialize GLFW")
        return -1

    window = glfw.create_window(1000, 1000, "KU-LOGO", None, None)
    if not window:
        glfw.terminate()
        return -1

    glfw.make_context_current(window)

    while not glfw.window_should_close(window):
        glClearColor(1.0, 1.0, 1.0, 1.0)
        glClear(GL_COLOR_BUFFER_BIT)
        glMatrixMode(GL_PROJECTION)
        glLoadIdentity()
        glOrtho(0, x_center*2 ,y_center*2, 0, 0, 1)
        glColor3f(1.0, 1.0, 1.0)

        red_polygon(x_center,y_center,360,12)
        divider_lines(x_center,y_center,360,12)
        draw_circle(x_center,y_center, 300, 360, black)
        draw_circle(x_center,y_center, 290, 360, blue)
        draw_circle(x_center,y_center, 230, 100, black)
        draw_circle(x_center,y_center, 220, 100, yellow)
        upsidedown_traingle(x_center,y_center,220,light_pink)
        traingle(x_center,y_center, 225, 185, white)
        draw_circle(x_center,375,30, 200,white)
        draw_circle(x_center-220* sqrt(3) / 2,y_center+220/2,30, 360,white)
        draw_circle(x_center+220* sqrt(3) / 2,y_center+220/2,30, 200,white)
        draw_rectangle(x=540,y=525,width=20,height=135)
        draw_u_shape(x=605,y=525,width=20,height=80,space=50,rotation=30)

        glfw.swap_buffers(window)
        glfw.poll_events()

    glfw.terminate()

if __name__ == "__main__":
    main()
```

Hence in this way we were able to achieve the logo of KU with different color and shapes drawing .