B22CS023
Assignment-0

Que1) 3D Rotating Cube

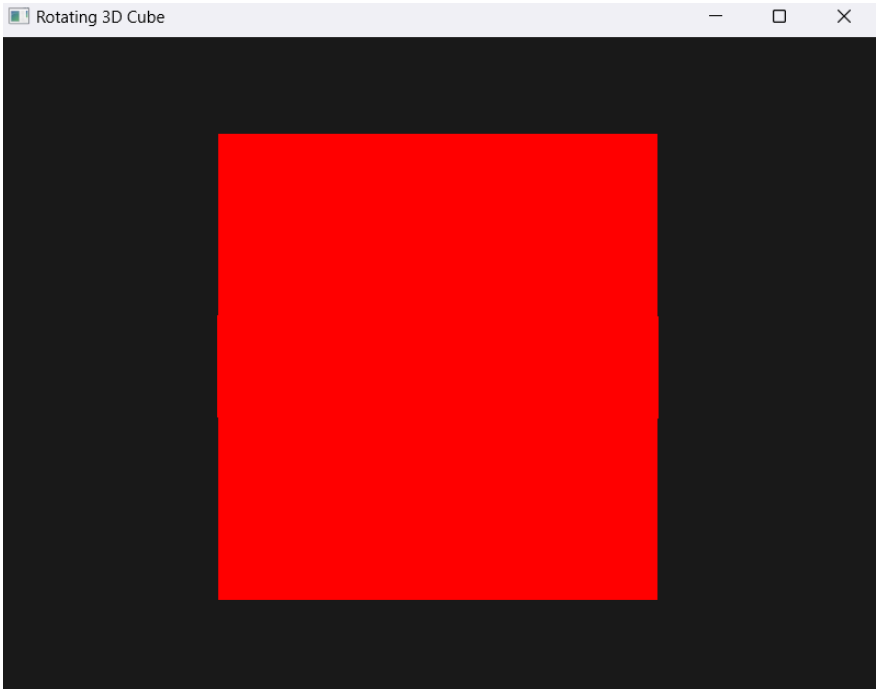
The task was to create a 3D object (here a cube is chosen), make it rotate around its axes, and keep things simple with a single color for all vertices.

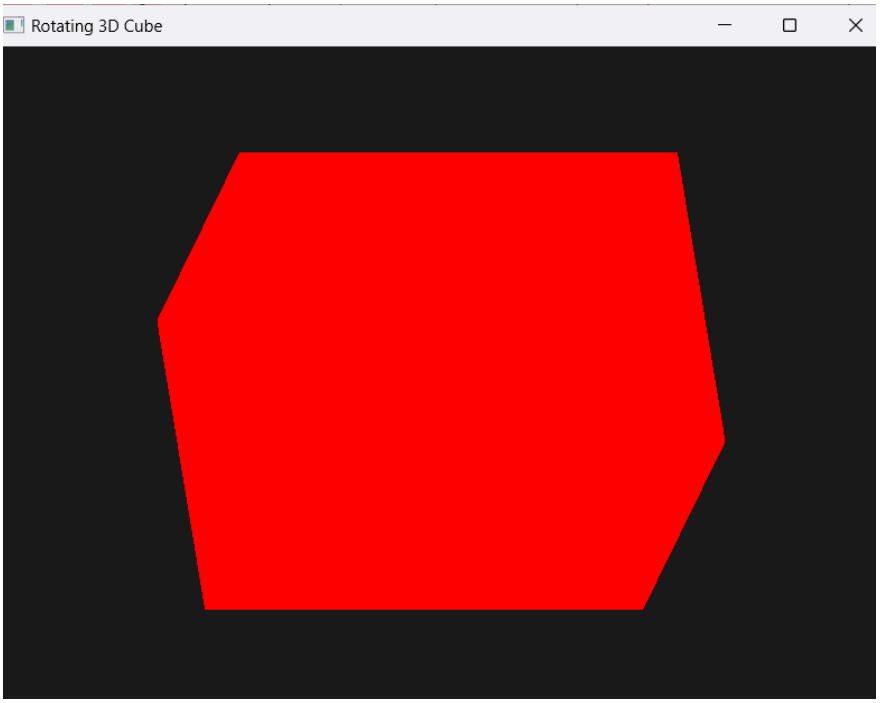
Working:

1. **Libraries and Tools**:
   - ➢ **GLFW**: Used for creating the window and handling events like resizing and closing.
   - ➢ **PyOpenGL**: For rendering the 3D cube and interacting with OpenGL functions.
   - ➢ **NumPy**: Managed the vertices and indices in a structured format.
   - ➢ **Pyrr**: Simplified the creation of rotation matrices for smooth cube animations.
2. **Shaders**:
   - ➢ **Vertex Shader:** Manages the cube's vertex placement and applies the rotation matrix. It also passes the vertex color to the fragment shader.
   - ➢ **Fragment Shader**: Focuses on displaying the color for each fragment on the cube's surface.
3. **Cube Creation**:
   The cube was defined using a set of vertices and indices. These explain where the cube's corners are located and how they are joined to create its faces. A single color was used across all faces for simplicity.
4. **Rotation Logic**:
   Rotations along the X and Y axes were implemented using matrices. These matrices were updated each frame using the elapsed time (`glfw.get_time()`), resulting in smooth, continuous spinning of the cube.
5. **Rendering Loop**:
   The rendering loop handles screen updates by clearing the display, applying the updated rotation matrix, and rendering the cube again in each frame. This continues until the window is closed.

The Output:

The end product is a 3D cube that rotates on both the X and Y axes and has a pleasing appearance. The rotation is smooth, and the single color highlights the simplicity of the object.


Rotating 3D Cube


Rotating 3D Cube

Que2) 2D Scenery with OpenGL: Grass, River, Mountains, and a Sunset

This program creates a simple 2D scene using OpenGL. The scene includes green grass, a flowing blue river, towering brown mountains, and a vibrant sunset nestled between the peaks.

Working:

1. **Grass**
   - ➢ A green rectangle at the bottom of the screen represents the grass.
   - ➢ This was drawn using a **GL_QUADS** primitive and carefully positioned vertices to span the bottom portion of the window.
2. **River**
   - ➢ A blue rectangle is drawn above the grass to represent a river.
   - ➢ The river adds depth to the scene and creates a natural separation between the grass and the mountains.
3. **Mountains**
   - ➢ Two brown triangles form the mountains.
   - ➢ Each triangle was defined using **GL_TRIANGLES** with coordinates ensuring symmetry and alignment with the sun's position.
   - ➢ These triangles add height and structure to the scenery.
4. **Sunset**
   - ➢ The sunset is represented as a partial arc positioned between the two mountain peaks.
   - ➢ The arc is created by calculating vertices for the sun’s visible portion using trigonometry.
   - ➢ A gradient of colors (yellow to red) was applied to simulate a natural sunset.
5. **Background Sky**
   - ➢ The background is set to a light blue color to give the impression of a clear daytime sky.
   - ➢ This was achieved using `glClearColor` in the initialization phase.

Rendering Flow:

- The `render_scene` function orchestrates the drawing process.
- Starting from the bottom layer (grass) and moving upward (river, mountains, and sun), each element is drawn in order.
- The layering ensures that the sun appears behind the river and mountains, enhancing realism.

Challenges:

- **Sunset Arc Calculation**
  The sunset is not a full circle but only the part visible between the mountain peaks. This was done by calculating the angle between the two mountain peaks and generating vertices for just that section of the circle. The use of trigonometry and polar coordinates simplified this process.

The Output:

The result is 2D landscape with:

- **Green grass** at the bottom.
- A **blue river** running horizontally.
- **Two brown mountains** with a vibrant **sunset** in the "V" shape between them.