- Lab 04 – OpenGL Drawing and Practical based Viva

Practical based Viva

You will be tested on the basics of OpenGL during this practical session. The activities contained in the following sheet are to be completed and you can practice them.

In the Viva session you will be required to modify the code, define functions, and explain the resulting program behavior. This evaluation will be conducted as an individual viva.

Submissions : Submit the code and screenshots of the outputs of Activity 01 and Activity 02

OpenGL related libraries

- OpenGL Utility Library (GLU)
- OpenGL Utility Toolkit (GLUT)
    - OpenSource version: FreeGLUT
- Graphical Library Framework

(GLFW) OpenGL Utility Toolkit

(GLUT) Window Management

- glutInit(int *argc, char **argv):
    - Initializes GLUT and processes any command-line arguments
- glutInitDisplayMode(unsigned int mode):
    - Specify the display colour and buffer models
- glutInitWindowPosition(int x, int y):
    - specifies the screen location for the upper-left corner of your window.
- glutInitWindowSize(int width, int height):
    - specifies the size, in pixels, of your window.
- glutCreateWindow(char *string):
    - creates a window with an OpenGL context.

Clearing Buffers

- glClear(GL_COLOR_BUFFER_BIT): Clearing window (colour buffer)

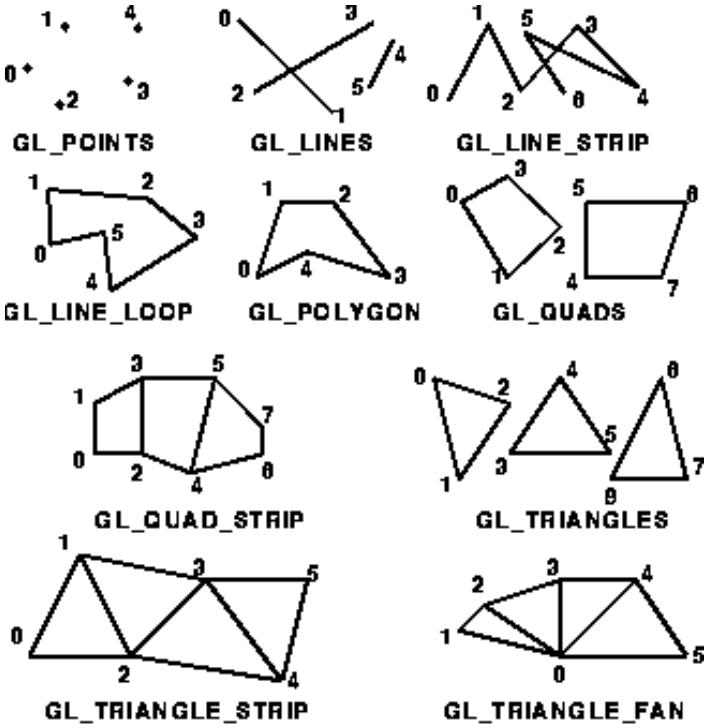
The Display callback

- glutDisplayFunc(void (*func)(void)):
    - Call the routine needs to draw the scene

Running the Program

- glutMainLoop(): Show all the windows that have been created and render the context nto the active window.

OpenGL Drawing Primitives



Object Creation

- glBegin() and glEnd(), define the object to be drawn.
- glVertex3f(float x,y,z) is used to define the position of a vertex in the object.

Setting a Colour

- glColor3f() command establishes what colour to use for drawing objects.

Sample Code

```
#include <GL/glut.h>
#include <stdlib.h>
void display(void) { // Display callback function
    glClear(GL_COLOR_BUFFER_BIT);     // Clear the color buffer (window)
    // Draw a red triangle
    glColor3f(1.0, 0.0, 0.0); // Red color
    glBegin(GL_TRIANGLES);
        glVertex3f(-0.6, -0.4, 0.0); // Bottom left vertex
        glVertex3f(0.6, -0.4, 0.0);  // Bottom right vertex
        glVertex3f(0.0, 0.6, 0.0);   // Top vertex
    glEnd();
    // Draw a green square
    glColor3f(0.0, 1.0, 0.0); // Green color
    glBegin(GL_QUADS);
        glVertex3f(-0.8, -0.8, 0.0); // Bottom left
        glVertex3f(-0.3, -0.8, 0.0); // Bottom right
        glVertex3f(-0.3, -0.3, 0.0); // Top right
        glVertex3f(-0.8, -0.3, 0.0); // Top left
    glEnd();
    // Draw a blue pentagon
    glColor3f(0.0, 0.0, 1.0); // Blue color
    glBegin(GL_POLYGON);
        glVertex3f(0.3, -0.8, 0.0);
        glVertex3f(0.6, -0.8, 0.0);
        glVertex3f(0.7, -0.5, 0.0);
        glVertex3f(0.5, -0.2, 0.0);
        glVertex3f(0.2, -0.2, 0.0);
    glEnd();
    // Flush the drawing commands
    glFlush();
}
int main(int argc, char** argv) {
   glutInit(&argc, argv); // Initialize GLUT and process command-line arguments
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);// Set display mode with RGB color
and single buffer
    glutInitWindowPosition(100, 100);// Set window position (upper-left corner at
100, 100)
    glutInitWindowSize(500, 500); // Set window size (500x500 pixels)
    glutCreateWindow("OpenGL GLUT Example"); // Create window with title
    glClearColor(0.0, 0.0, 0.0, 1.0); // Set the clear color (background color) to
black
    glutDisplayFunc(display); // Register display callback function
    glutMainLoop(); // Enter the GLUT main loop
    return 0;
}
```

Activity 1: Drawing a Cube

(Following code goes with in main method)

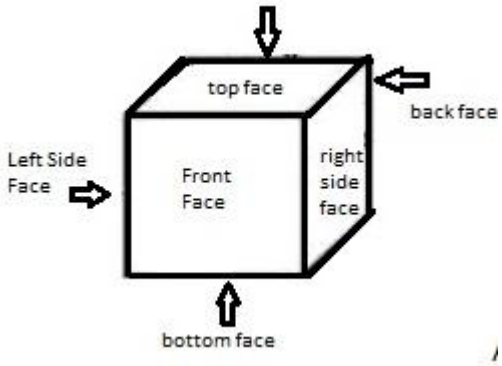
1. Create a window with name " Cube of (YourITIndex) "
2. Draw a window of size 500px wide and 500px in height.
3. Position the window at x = 200 px and y =200 px
4. Create a separate function named createCube.
5. Code following within the createCube function.
    a. Make the background color of the window black.
    b. Clear the color buffer by using glClear(mask name) function.

    GL_COLOR_BUFFER_BIT is the mask that represents the buffers that are currently being used for applying colors
    c. glBegin(mode) used for grouping statements that lead to a specific shape. You can create different shapes such as points, lines, triangles, quads, and more, by grouping the required vertices within this grouping statement. The shape that you want to create can be specified by specifying any of the modes. glBegin(mode) ends with glEnd().
    d. Create six groups of glBegin() to form six faces of a cube by defining vertices. You may use GL_POLYGON as the mode. Use the function glVertex3f(x,y,z) to define vertices and glColor3f(R,G,B) to set color to the polygons.

    Set colors of the cube as follows:
    - Front: Red
    - Back: Green
    - Left and Right: Blue
    - Top: White
    - Bottom: yellow

    Coordinates (x,y,z) of the vertices of front face are:
    - Bottom left: -0.4, -0.2, 0
    - Top left: -0.4, 0.2, 0
    - Top right: 0, 0.2, 0
    - Bottom right: 0, -0.2, 0

    Coordinates (x,y,z) of the vertices of back face are:
    - Bottom left: -0.2, 0, -0.4
    - Top left: -0.2, 0.4, -0.4
    - Top right: 0.2, 0.4, -0.4
    - Bottom right: 0.2, 0, -0.4

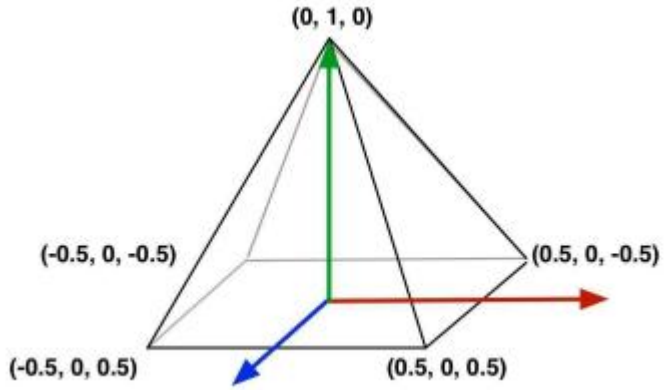


    e. call glFlush() function execute code without waiting others to complete.

Activity 2: Drawing a pyramid

a. Using the same techniques used in the above activity try to draw a pyramid (use any color according to your preference).

Coordinates of the vertices are given by the following diagram



You may use GL_TRIANGLES completely or a combination of GL_TRIANGLES with GL_POLYGON or GL_QUADS. Hint: you can form a square shape by using two triangles

Evaluation Rubric

Criteria	Max marks	Good (marks 2)	Satisfactory (marks 1)	Poor (Marks 0)
1. Window Management *Change window size and position*	2	Correctly uses glutInitWindowSize() and glutInitWindowPosition() with appropriate values. Changes are demonstrated and functional.	Uses only one function correctly, or both with significant errors (e.g., values cause window to be off-screen).	Unable to use the functions and syntax is incorrect, or values are nonsensical.
2. Window Creation *Create window with a title*	2	Correctly uses glutCreateWindow() with a string argument. The window title is personalized (e.g., includes IT Index).	Uses glutCreateWindow() but with a generic or empty title string.	Does not use glutCreateWindow(), or the call is incorrect. Window does not open.
3. Background Setup *Set the background color*	2	Correctly uses glClearColor() to set an RGBA color (e.g., black) and glClear(GL_COLOR_BUFFER_BIT) in the display function.	Uses glClearColor() but forgets to call glClear(), or vice-versa. Background is default or undefined.	Unable to set a background color.
4. Primitive Manipulation *Change positions/sizes of primitives*	2	Demonstrates clear understanding by modifying vertex coordinates (glVertex3f) to successfully change the position, size, and shape of drawn objects (e.g., cube, pyramid).	Modifies coordinates but with significant errors, creating an unrecognizable or incorrect shape.	Unable to modify values but uses incorrect function or syntax.
5. Advanced Transformation ***Change color, transform objects***	2	Successfully changes colors of individual faces (glColor3f) and demonstrates ability to "transform" an object (e.g., cube to a rectangular prism, pyramid to a tetrahedron) by logically altering its vertex structure.	Changes only the overall color of the object. Attempts to change the object but new object is incorrect.	Unable to change the color or object