Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

Lab Report

[COMP 342]

Submitted by

Soniya Sharma(47)

Submitted to

Dhiraj Shrestha
Department of Computer Science and Engineering

Submission Date:

2024-04-24

Contents

1 Mention the name of Programming language and Graphics Library you are using this semester for performing your Computer Graphics Lab and Project. **1**

2 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 **1**

3 ScreenShot of Source Code **2**

4 Output **8**

5 Conclusion **8**

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

The programming language used for performing Computer Graphics Lab is C++. The graphics library utilized is OpenGL (Open Graphics Library).In addition to C++ and OpenGL, the GLFW (Graphics Library Framework) is also used in the provided code snippets. GLFW is a C library that provides a simple API for creating windows, contexts, and handling input events such as keyboard and mouse interactions.

2 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

```
#include <iostream>
#include <GLFW/glfw3.h>
#include <cmath>
```

.vscode > {} tasks.json > [ ] tasks > {} 0 > [ ] args

```
{
    "version": "2.0.0",
    "tasks": [
     {
      "type": "cppbuild",
      "label": "C/C++: clang++ build active file",
      "command": "/usr/bin/clang++",
      "args": [
       "-std=c++17",
       "-fdiagnostics-color=always",
       "-Wall",
       "-g",
       "-I${workspaceFolder}/dependencies/include",
       "-L${workspaceFolder}/dependencies/library",
       "${workspaceFolder}/dependencies/library/libglfw.3.4.dylib",
       "${workspaceFolder}/*.cpp",
       "${workspaceFolder}/glad.c",
       "-o",
       "${workspaceFolder}/app",
       "-framework",
       "OpenGL",
       "-framework",
       "Cocoa",
       "-framework",
       "IOKit",
       "-framework",
       "CoreVideo",
       "-framework",
       "CoreFoundation",
       "-Wno-deprecated"
      ],
      "options": {
       "cwd": "${fileDirname}"
      },
      "problemMatcher": ["$gcc"],
      "group": {
       "kind": "build",
       "isDefault": true
      },
      "detail": "compiler: /usr/bin/clang++"
     }
    ]
}
```

```
soniyasharma@Soniyas-MacBook-Air youtube %
Resolution of the display system: 1440x900
```

3 ScreenShot of Source Code

```cpp
#include <iostream>
#include <GLFW/glfw3.h>
#include <cmath>

#define PI 3.1415f

using namespace std;

void errorCallback(int error, const char *description);
void framebuffer_size_callback(GLFWwindow *window, int width, int height);

void drawMountain();
void drawMountainWithSnow();
void filloutsideMountain();
void drawSquare();
void drawDome();
void ntb_text();
void drawTempleRoof();
void drawtemple();
void drawRoofbar();
void drawDomeHead();
void drawRedpoly();
void smalltriangle();
void drawSnow();
void drawFace();
```

(a) CodeSnipet 1

```cpp
int main()
{
    if (!glfwInit())
    {
        cout << "Failed to initialize GLFW" << endl;
        return -1;
    }
    glfwSetErrorCallback(errorCallback);

    GLFWwindow *window = glfwCreateWindow(1000, 1000, "NTB Logo", NULL, NULL);
    if (!window)
    {
        cout << "Failed to create a window" << endl;
        glfwTerminate();
        return -1;
    }

    glfwMakeContextCurrent(window);

    glfwSetFramebufferSizeCallback(window, framebuffer_size_callback);

    while (!glfwWindowShouldClose(window))
    {
        glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT);
        drawMountain();
        drawMountainWithSnow();
        drawSnow();
        glLineWidth(40.0f);

        filloutsideMountain();
        drawDomeHead();
        drawRedpoly();
        drawSquare();
        drawDome();
        drawtemple();
```

(b) Code Snippet 2

```cpp
// Function to draw a mountain with snow overlay
void drawMountainWithSnow()
{
    // Save the current transformation state
    glPushMatrix();

    // Translate the mountain and snow overlay to the desired position
    glTranslatef(-0.4f, 0.25f, 0.0f);

    // Set the line width for drawing
    glLineWidth(40.0f);

    // Set the color for drawing the mountain and snow overlay
    glColor3f(0.2039f, 0.2353f, 0.5765f);

    // Draw the mountain base
    drawMountain();

    // Begin drawing the snow overlay polygon
    glBegin(GL_POLYGON);
    // Set the color for the snow overlay
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    // Define vertices of the snow overlay polygon
    glVertex2f(0.75f, -0.1f);
    glVertex2f(-0.4f, -1.0f);
    glVertex2f(1.7f, -1.0f);
    // End drawing the snow overlay polygon
    glEnd();

    // Restore the previous transformation state
    glPopMatrix();
}
```

(c) Code Snippet 3

```cpp
        ntb_text();


        glfwSwapBuffers(window);
        glfwPollEvents();
    }

    glfwTerminate();
    return 0;
}

void errorCallback(int error, const char *description)
{
    cout << "Error: " << description << endl;
}

void framebuffer_size_callback(GLFWwindow *window, int width, int height)
{
    glViewport(0, 0, width, height);
}

// Function to draw the base polygon representing a mountain
void drawMountain()
{
    // Begin drawing the polygon
    glBegin(GL_POLYGON);
    // Define vertices of the polygon
    glVertex2f(0.75f, -0.1f);
    glVertex2f(-0.25f, -1.0f);
    glVertex2f(1.75f, -1.0f);
    glVertex2f(0.75f, -0.1f);
    // End drawing the polygon
    glEnd();
}
```

(d) Code Snippet 4

```cpp
// Function to draw snowflakes
void drawSnow()
{
    // Save the current transformation state
    glPushMatrix();

    // Begin drawing the snowflake polygon
    glBegin(GL_POLYGON);

    // Set the color for the snowflake (white)
    glColor3f(1.0f, 1.0f, 1.0f);

    // Define vertices of the snowflake polygon in anticlockwise order
    glVertex2f(0.35f, 0.14f);   // First vertex
    glVertex2f(0.235f, 0.05f);  // Second vertex
    glVertex2f(0.29f, -0.01f);  // Third vertex
    glVertex2f(0.34f, 0.05f);   // Fourth vertex
    glVertex2f(0.37f, 0.03f);   // Fifth vertex
    glVertex2f(0.40f, 0.05f);   // Sixth vertex
    glVertex2f(0.485f, -0.01f); // Seventh vertex
    glVertex2f(0.49f, 0.01f);   // Eighth vertex

    // End drawing the snowflake polygon
    glEnd();

    // Restore the previous transformation state
    glPopMatrix();
}
```

(e) Code Snippet 5

```cpp
void filloutsideMountain()
{
    // filling white inside outside mountain
    glPushMatrix();
    glTranslatef(0.0f, -0.07f, 0.0f);
    glLineWidth(40.0f);
    glColor3f(0.2039f, 0.2353f, 0.5765f);

    drawMountain();
    glBegin(GL_POLYGON);
    glColor3f(1.0f, 1.0f, 1.0f);
    glVertex2f(0.75f, -0.1f);
    glVertex2f(-0.4f, -1.0f);
    glVertex2f(1.7f, -1.0f);
    glEnd();
    glPopMatrix();
}
```

(f) Code Snippet 6

```
void drawSquare()
Click to collapse the range.
    glLineWidth(25.0f);
    // top orange
    glBegin(GL_LINE_STRIP);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    glVertex2f(-0.035f, 0.15f);
    glVertex2f(-0.465f, 0.15f);
    // glVertex2f(-0.035f, 0.19f);
    // glVertex2f(-0.465f, 0.19f);
    glEnd();

    glLineWidth(20.0f);

    glBegin(GL_LINE_STRIP);
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glVertex2f(-0.08f, 0.115f);
    glVertex2f(-0.08f, -0.4f);
    glEnd();

    glBegin(GL_LINE_STRIP);
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glVertex2f(-0.425f, 0.115f);
    glVertex2f(-0.425f, -0.165f);
    glEnd();

    // inside of square
    drawFace();
}
```

(g) Code Snippet 7

```
void drawFace()
{
    glColor3f(0.8863f, 0.0f, 0.1333f);

    glBegin(GL_TRIANGLE_FAN);
    float centrex = -0.25f;
    float centrey = 0.075f;
    for (int i = 0; i <= 100; ++i)
    {
        float angle = 2.0f * PI * float(i) / float(100);
        float x = centrex + 0.015f * cos(angle);
        float y = centrey + 0.015f * sin(angle);
        glVertex2f(x, y);
    }
    glEnd();
```

(h) Code Snippet 8

```
    //left eye
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glLineWidth(5.0f); // Set line width
    glBegin(GL_LINE_LOOP);
    glVertex2f(-0.365f, 0.0f);  // Bottom-left vertex
    glVertex2f(-0.385f, 0.0175f);
    glVertex2f(-0.365f, 0.035f); // Top-left vertex
    glVertex2f(-0.3f, 0.035f);   // Top-right vertex
    glVertex2f(-0.28f, 0.0175f);
    glVertex2f(-0.3f, 0.0f);     // Bottom-right vertex
    glEnd();
    //left eyeball
    glBegin(GL_TRIANGLE_FAN);
    float eyeCentrex = -0.3325f;
    float eyeCentrey = 0.0175f;
    for (int i = 0; i <= 100; ++i)
    {
        float angle = 2.0f * PI * float(i) / float(100);
        float x = eyeCentrex + 0.015f * cos(angle);
        float y = eyeCentrey + 0.015f * sin(angle);
        glVertex2f(x, y);
    }
    glEnd();
    //// eyebrows
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glBegin(GL_LINE_STRIP);
    glVertex2f(-0.385f, 0.0475f);
    glVertex2f(-0.365f, 0.065f);
    glVertex2f(-0.305f, 0.065f);
    glVertex2f(-0.28f, 0.0475f);
    glEnd();
```

(i) Code Snippet 9

```
    //right eye
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glLineWidth(5.0f); // Set line width
    glBegin(GL_LINE_LOOP);
    glVertex2f(-0.20f, 0.0f);    // Bottom-left vertex
    glVertex2f(-0.22f, 0.017f);
    glVertex2f(-0.20f, 0.035f);   // Top-left vertex
    glVertex2f(-0.14f, 0.035f);   // Top-right vertex
    glVertex2f(-0.12f, 0.017f);
    glVertex2f(-0.14f, 0.0f);     // Bottom-right vertex
    glEnd();
    //right eyeball
    glBegin(GL_TRIANGLE_FAN);
    eyeCentrex = -0.17f;
    eyeCentrey = 0.0175f;
    for (int i = 0; i <= 100; ++i)
    {
        float angle = 2.0f * PI * float(i) / float(100);
        float x = eyeCentrex + 0.015f * cos(angle);
        float y = eyeCentrey + 0.015f * sin(angle);
        glVertex2f(x, y);
    }
    glEnd();
    //eyebrows
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glBegin(GL_LINE_STRIP);
    glVertex2f(-0.22f, 0.0475f);
    glVertex2f(-0.2f, 0.065f);
    glVertex2f(-0.14f, 0.065f);
    glVertex2f(-0.12f, 0.0475f);
    glEnd();
```

(j) Code Snippet 10

```
    ////nose
    glBegin(GL_LINE_STRIP);
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    glLineWidth(1.0f);
    for (int i = 100 * 2.25; i >= 0; --i)
    {
        float theta = -2.0f * PI * i / float(100);
        float x = -0.25f + (0.001f + i * 0.00015f) * cos(theta);
        float y = -0.035 + (0.001f + i * 0.00015f) * sin(theta);
        glVertex2f(x, y);
    }
    glEnd();
}

void drawtemple()
{
    drawTempleRoof();
    glPushMatrix();
    glScalef(1.5f, 1.2f, 1.0f);
    glTranslatef(0.17f, -0.3f, 0.0f);
    drawTempleRoof();
    glPopMatrix();

    drawRoofbar();
    // smaller triangle
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    smalltriangle();
}
```

(k) Code Snippet 11

```
void smalltriangle()
{
    glPushMatrix();
    glTranslatef(-0.54f, 0.11f, 0.0f);
    glLineWidth(40.0f);
    glScalef(0.03f, 0.07f, 1.0f);
    drawMountain();
    glPopMatrix();
}

void drawTempleRoof()
{
    glColor3f(0.2039f, 0.2353f, 0.5765f);

    glBegin(GL_QUADS);
    glVertex2f(-0.8f, -0.165f);  // Bottom-left vertex
    glVertex2f(-0.25f, -0.165f); // Bottom-right vertex
    glVertex2f(-0.451f, 0.015f);  // Top-right vertex
    glVertex2f(-0.6f, 0.015f);    // Top-left vertex
    glEnd();

    glLineWidth(25.0f);
    // top orange
    glBegin(GL_LINE_STRIP);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    glVertex2f(-0.24f, -0.20f);
    glVertex2f(-0.82f, -0.20f);
    glEnd();
}
```

(l) Code Snippet 12

```
void drawRoofbar()
{
    glLineWidth(25.0f);
    // top orange
    glColor3f(0.8863f, 0.0f, 0.1333f);
    glBegin(GL_QUADS);
    glVertex2f(-0.63f, -0.32f);  // Bottom-left vert
    glVertex2f(-0.42f, -0.32f);  // Bottom-right ver
    glVertex2f(-0.3f, -0.20f);   // Top-right vertex
    glVertex2f(-0.75f, -0.20f);  // Top-left vertex
    glEnd();

    glColor3f(1.0f, 1.0f, 1.0f);

    glBegin(GL_QUADS);
    glVertex2f(-0.61f, -0.29f); // Bottom-left verte
    glVertex2f(-0.44f, -0.29f); // Bottom-right vert
    glVertex2f(-0.36f, -0.22f); // Top-right vertex
    glVertex2f(-0.69f, -0.22f); // Top-left vertex
    glEnd();

    // titled triangles
    glPushMatrix();
    glTranslatef(-0.4f, 0.14f, 0.0f);
    glLineWidth(40.0f);
    glRotatef(45.0f, 0.0f, 0.0f, 1.0f);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    smalltriangle();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(0.08f, -0.59f, 0.0f);
    glLineWidth(40.0f);
```

(m) Code Snippet 13

```
    glPushMatrix();
    glTranslatef(0.08f, -0.59f, 0.0f);
    glLineWidth(40.0f);
    glRotatef(-45.0f, 0.0f, 0.0f, 1.0f);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    smalltriangle();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(-0.55f, -0.26f, 0.0f);
    glLineWidth(40.0f);
    glRotatef(45.0f, 0.0f, 0.0f, 1.0f);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    smalltriangle();
    glPopMatrix();

    glPushMatrix();
    glTranslatef(0.205f, -0.99f, 0.0f);
    glLineWidth(40.0f);
    glRotatef(-45.0f, 0.0f, 0.0f, 1.0f);
    glColor3f(0.8863f, 0.0f, 0.1333f);
    smalltriangle();
    glPopMatrix();
}
```

(n) Code Snippet 14

```
void drawDomeHead()
{
    glPushMatrix();
    glTranslatef(-0.37f, 0.94f, 0.0f);
    glLineWidth(40.0f);
    glScalef(0.16f, 0.74f, 1.0f);
    glColor3f(0.2039f, 0.2353f, 0.5765f);

    drawMountain();
    glPopMatrix();

    // for white strips
    float x_coord1 = -0.035f;
    float x_coord2 = -0.465f;
    float y_coord = 0.24f;
    float line_width = 9.7f;
    for (int i = 0; i < 13; i++)
    {
        glLineWidth(line_width);
        glBegin(GL_LINE_STRIP);
        glColor3f(1.0f, 1.0f, 1.0f);
        glVertex2f(x_coord1, y_coord);
        glVertex2f(x_coord2, y_coord);
        glEnd();
        y_coord = y_coord + 0.05f;
        line_width = line_width + 0.4f;
    }
```

(o) Code Snippet 15

```
    // little triangle
    glPushMatrix();
    glTranslatef(-0.268f, 0.95f, 0.0f);
    glLineWidth(40.0f);
    glScalef(0.025f, 0.13f, 1.0f);
    glColor3f(0.2039f, 0.2353f, 0.5765f);
    drawMountain();
    glPopMatrix();

    glColor3f(0.2039f, 0.2353f, 0.5765f);
    // little blue line

    glLineWidth(18.0f);
    glBegin(GL_LINE_STRIP);
    glVertex2f(-0.23f, 0.81);   // top-left
    glVertex2f(-0.265f, 0.81);  // top-right
    glEnd();

    glLineWidth(18.0f);
    glBegin(GL_LINE_STRIP);
    glVertex2f(-0.207f, 0.82);   // top-left
    glVertex2f(-0.290f, 0.82);  // top-right
    glEnd();
}
```

(p) Code Snippet 16

```
void drawRedpoly()
{
    //outer white
    glColor3f(1.0f, 1.0f, 1.0f);
    glBegin(GL_POLYGON);
    glVertex2f(-0.155f, 0.15f);  // Bottom-right
    glVertex2f(-0.35f, 0.15f);   // Bottom-left
    glVertex2f(-0.35f, 0.23f);   // Top-left
    glVertex2f(-0.255f, 0.30f);  // Top-middle
    glVertex2f(-0.155f, 0.23f);  // Top-right
    glEnd();

    //red
    glColor3f(0.8863f, 0.0f, 0.1333f);
    glBegin(GL_POLYGON);
    glVertex2f(-0.17f, 0.15f);  // Bottom-right
    glVertex2f(-0.34f, 0.15f);  // Bottom-left
    glVertex2f(-0.34f, 0.2f);   // Top-left
    glVertex2f(-0.255f, 0.28f);  // Top-middle
    glVertex2f(-0.17f, 0.2f);   // Top-right
    glEnd();

    //inner white
    glBegin(GL_POLYGON);
    glColor3f(1.0f, 1.0f, 1.0f);
    glVertex2f(-0.19f, 0.15f);  // Bottom-right
    glVertex2f(-0.32f, 0.15f);  // Bottom-left
    glVertex2f(-0.255f, 0.24f);  // Top-middle
    glEnd();
}
```

(q) Code Snippet17

```
    void drawDome()
{
    float cx = -0.3f;
    float cy = -1.0f;
    float r = 0.8f;
    int num_segments = 180;

    glColor3f(0.2039f, 0.2353f, 0.5765f);
    // glLineWidth(70.0f);
    glBegin(GL_LINE_STRIP);
    for (int i = 0; i <= num_segments / 2; i++)
    {
        float theta = PI * float(i) / float(num_segments);
        float x = r * cosf(theta);
        float y = r * sinf(theta);
        glVertex2f(x + cx, y + cy);
    }
    glEnd();

    glColor3f(1.0f, 1.0f, 1.0f);
    glBegin(GL_POLYGON);
    for (int i = 0; i <= num_segments / 2; i++)
    {
        float theta = 2 * PI * float(i) / float(num_segments);
        float x = r * cosf(theta);
        float y = r * sinf(theta);
        glVertex2f(x + cx, y + cy);
    }
    glEnd();
}
```

(r) Code Snippet 18

```
void ntb_text(){
    // Set color for the text
    glColor3f (0.2039f, 0.2353f, 0.5765f);

    // Draw letter 'N'

    glBegin(GL_LINES);
    glVertex2f(-0.935f, -0.66f); // Start of left vertical line
    glVertex2f(-0.935f, -0.88f); // End of left vertical line
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(-0.935f, -0.668f); // Start of diagonal line (common point)
    glVertex2f(-0.7225f, -0.874f); // End of diagonal line
    glEnd();

    glBegin(GL_LINES);

    glVertex2f(-0.7225f, -0.66f); //  Start of right vertical line
    glVertex2f(-0.7225f, -0.881f); // End of right vertical line
    glEnd();



    // Draw letter 'T'
    glBegin(GL_LINES);

    glVertex2f(-0.35f, -0.675f); // Start of top vertical line
    glVertex2f(-0.35f, -0.865f); // End of top vertical line
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(-0.2f, -0.67f); // Start of horizontal line
```

(s) Code Snippet 19

```
    glVertex2f(-0.5f, -0.67f); // End of horizontal line
    glEnd();


    // Draw letter 'B'
    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.66f); // Start of first vertical
    glVertex2f(0.0f, -0.88f); // End of first vertical li
    glEnd();


    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.67f); // Start of top horizontal
    glVertex2f(0.2f, -0.67f); // End of top horizontal li
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.77f); // Start of middle horizont
    glVertex2f(0.2f, -0.77f); // End of middle horizontal
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.870f); // Start of bottom horizon
    glVertex2f(0.2f, -0.870f); // End of bottom horizonta
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.2f, -0.66f); // Start of right vertical
    glVertex2f(0.2f, -0.88f); // End of right vertical li
    glEnd();
```

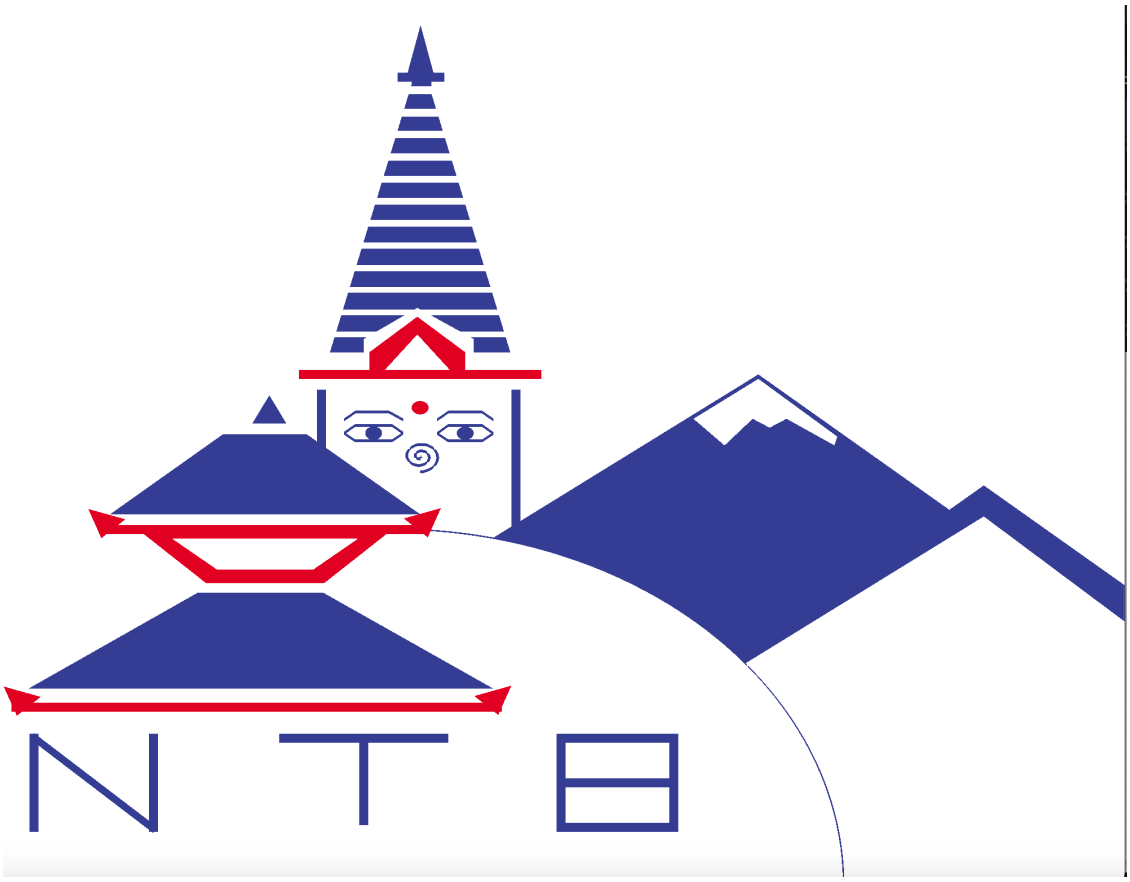
(t) Code Snippet 20

```
    glVertex2f(-0.5f, -0.67f); // End of horizontal line
    glEnd();


    // Draw letter 'B'
    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.66f); // Start of first vertical line
    glVertex2f(0.0f, -0.88f); // End of first vertical line
    glEnd();


    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.67f); // Start of top horizontal line
    glVertex2f(0.2f, -0.67f); // End of top horizontal line
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.77f); // Start of middle horizontal line
    glVertex2f(0.2f, -0.77f); // End of middle horizontal line
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.0f, -0.870f); // Start of bottom horizontal line
    glVertex2f(0.2f, -0.870f); // End of bottom horizontal line
    glEnd();

    glBegin(GL_LINES);
    glVertex2f(0.2f, -0.66f); // Start of right vertical line
    glVertex2f(0.2f, -0.88f); // End of right vertical line
    glEnd();
}
```

(u) Code Snippet 21

4 Output

The output of the source code is logo of Nepal Tourism Board(NTB).



(v) Output Window

5 Conclusion

While doing this lab, we become familiar with OPENGL and GLFW library. Learn how to render and generate shapes and symbols.