**Lab Taks-4**

Submission Guidelines-

* Rename the file with your serial number only
* Must submit within time that will be discussed in class VUES
* Must include resources for all the section in the table
* *Question-3 is optional. There is no problem if you do not implement it.*

|  |
| --- |
| **Question- 1**  Design the given scenario |
| **Graph Plot (Picture)-** |
| **Code-#include <windows.h> // for MS Windows**  **#include <GL/glut.h>**  **#include <cmath> // GLUT, include glu.h and gl.h**  **void circle(float radius, float xc, float yc, float r, float g, float b)**  **{**  **glBegin(GL\_LINE\_LOOP);// Draw a Red 1x1 Square centered at origin**  **for(int i=0; i<200; i++)**  **{**  **glColor3f(r,g,b);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=radius;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x+xc,y+yc);**  **}**  **glEnd();**  **}**  **void mountain()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glVertex2f(0,1);**  **glVertex2f(3.15,-2.7);**  **glVertex2f(-3.15,-2.7);**  **glEnd();**  **}**  **void Sun()**  **{**  **circle(2.2,0,1,0,0,0);**  **}**  **void Ground()**  **{**  **glBegin(GL\_LINES);**  **glVertex2f(5,-2.7);**  **glVertex2f(-5,-2.7);**  **glEnd();**  **}**  **void Lines()**  **{**  **glBegin(GL\_LINES);**  **glVertex2f(-3.04,1.9);**  **glVertex2f(-0.6,1.9);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(-2.57,0.8);**  **glVertex2f(-1.48,0.8);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(1.30755,1.6);**  **glVertex2f(2.39521,1.6);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(1.41632,0.8);**  **glVertex2f(2.3,0.8);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(-0.7,0);**  **glVertex2f(0,0);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(-1,-.8);**  **glVertex2f(-0.3,-.8);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(-2.5,-2);**  **glVertex2f(-0.5,-2);**  **glEnd();**  **glBegin(GL\_LINES);**  **glVertex2f(-2,-1.6);**  **glVertex2f(-0.3,-1.6);**  **glEnd();**  **}**  **void devide()**  **{**  **glBegin(GL\_LINES);**  **glVertex2f(0,1);**  **glVertex2f(0.4,-0.6);**  **glVertex2f(0.4,-0.6);**  **glVertex2f(0.0,-0.8);**  **glVertex2f(0.0,-0.8);**  **glVertex2f(-0.4,-1.2);**  **glVertex2f(-0.4,-1.2);**  **glVertex2f(0.0,-1.6);**  **glVertex2f(0.0,-1.6);**  **glVertex2f(-0.86776,-2.7);**  **glEnd();**  **}**  **void Tree1(){**  **glBegin(GL\_LINES);**  **glVertex2f(4,-1);**  **glVertex2f(4,-2.7);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(4,-1);**  **glVertex2f(4.6,-1.4);**  **glVertex2f(3.4,-1.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(4,-1.2);**  **glVertex2f(4.6,-1.6);**  **glVertex2f(3.4,-1.6);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(4,-1.4);**  **glVertex2f(4.6,-1.8);**  **glVertex2f(3.4,-1.8);**  **glEnd();**  **}**  **void Tree2(){**  **glBegin(GL\_LINES);**  **glVertex2f(-4,-1);**  **glVertex2f(-4,-2.7);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(-4,-1);**  **glVertex2f(-4.6,-1.4);**  **glVertex2f(-3.4,-1.4);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(-4,-1.2);**  **glVertex2f(-4.6,-1.6);**  **glVertex2f(-3.4,-1.6);**  **glEnd();**  **glBegin(GL\_POLYGON);**  **glVertex2f(-4,-1.4);**  **glVertex2f(-4.6,-1.8);**  **glVertex2f(-3.4,-1.8);**  **glEnd();**  **}**  **void renderStrokeString(float x, float y, const char\* text, float scale) {**  **glPushMatrix(); // Save current state**  **glTranslatef(x, y, 0); // Move to position**  **glScalef(scale, scale, 1.0f); // Scale text**  **while (\*text) {**  **glutStrokeCharacter(GLUT\_STROKE\_ROMAN, \*text); // Render each character**  **text++;**  **}**  **glPopMatrix(); // Restore state**  **}**  **void Text(){**  **renderStrokeString(-5.0, -5.0, "MOUNTAIN", 0.015);**  **//renderBitmapString(-1.0f, -4.0f, "MOUNTAIN");**  **}**  **void display()**  **{**  **glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to white and opaque**  **glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)**  **glLineWidth(3);**  **// mountain();**  **Sun();**  **mountain();**  **Ground();**  **Lines();**  **devide();**  **Tree1(); Tree2();**  **Text();**  **glFlush(); // Render now**  **}**  **int main(int argc, char\*\* argv)**  **{**  **glutInit(&argc, argv); // Initialize GLUT**  **glutInitWindowSize(720, 720);**  **glutCreateWindow("Mountain"); // Create a window with the given title**  **// Set the window's initial width & height**  **glutDisplayFunc(display);**  **gluOrtho2D(-6,6,-6,8 );// Register display callback handler for window re-paint**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-A screenshot of a computer  Description automatically generated** |

|  |
| --- |
| **Question- 2**  Design the given scenario  Drawing Lighthouse Line Stock Illustrations – 1,639 Drawing Lighthouse Line  Stock Illustrations, Vectors & Clipart - Dreamstime |
| **Graph Plot (Picture)-**  A computer screen shot of a graph  Description automatically generated |
| **Code-**  **#include <windows.h> // for MS Windows**  **#include <GL/glut.h>**  **#include <cmath> // GLUT, include glu.h and gl.h**  **void Ground()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-4,-4);**  **glVertex2f(-0.97,-2.5);**  **glVertex2f(-0.0,-2.4);**  **glVertex2f(0.78,-2.5);**  **glVertex2f(4,-4);**  **glEnd();**  **}**  **void Ground2()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-6,-4);**  **glVertex2f(6,-4);**  **glEnd();**  **}**  **void Rectangle()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-0.14,-0.8);**  **glVertex2f(0.18,-0.8);**  **glVertex2f(0.23,-2.4);**  **glVertex2f(-0.2,-2.4);**  **glEnd();**  **}**  **void InnerRectangle1()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-0.0,-0.88);**  **glVertex2f(0.064,-0.88);**  **glVertex2f(0.078,-1.02);**  **glVertex2f(-0.018,-1.02);**  **glEnd();**  **}**  **void InnerRectangle2()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-0.0,-1.84);**  **glVertex2f(0.064,-1.84);**  **glVertex2f(0.078,-2.00);**  **glVertex2f(0.0,-2.0);**  **glEnd();**  **}**  **void TOPRectangle1()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(0.23,-0.55);**  **glVertex2f(-0.16,-0.55);**  **glVertex2f(-0.14,-0.8);**  **glVertex2f(0.18,-0.8);**  **glEnd();**  **}**  **void TOPT()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(0.023,-0.0998);**  **glVertex2f(-0.15,-0.31);**  **glVertex2f(0.21,-0.31);**  **glEnd();**  **}**  **void TOPR()**  **{**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(0,0,0);**  **glVertex2f(-0.085,-0.31);**  **glVertex2f(0.13,-0.31);**  **glVertex2f(0.13,-0.55);**  **glVertex2f(-0.093,-0.55);**  **glEnd();**  **}**  **void Lines(){**  **glBegin(GL\_LINES);**  **glColor3ub(0,0,0);**  **glVertex2f(0.023,0.03457);**  **glVertex2f(0.023,-0.0998);**  **glVertex2f(-0.18,-0.64);**  **glVertex2f(0.22,-0.64);**  **glVertex2f(-0.18,-0.695);**  **glVertex2f(0.22,-0.695);**  **glEnd();**  **}**  **void Top()**  **{**  **TOPR();**  **TOPT();**  **}**  **LineTOP(){**  **glBegin(GL\_LINES);**  **glColor3ub(0,0,0);**  **glVertex2f(-2.00857,-0.06395);**  **glVertex2f(-0.34323,-0.39303);**  **glVertex2f(-2.37753,-0.50);**  **glVertex2f(-0.37314,-0.50);**  **glVertex2f(-2,-1);**  **glVertex2f(-0.37314,-0.66481);**  **glVertex2f(0.45023,-0.37306);**  **glVertex2f(2.11641,-0.07483);**  **glVertex2f(0.5,-0.5);**  **glVertex2f(2.5,-0.5);**  **glVertex2f(0.48264,-0.65184);**  **glVertex2f(2.11641,-0.96952);**  **glEnd();**  **}**  **void display()**  **{**  **glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to white and opaque**  **glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)**  **glLineWidth(3);**  **Rectangle();**  **Ground();**  **Ground2();**  **InnerRectangle1();**  **InnerRectangle2();**  **TOPRectangle1();**  **Top();**  **Lines();**  **LineTOP();**  **glFlush();**  **// Render now**  **}**  **int main(int argc, char\*\* argv)**  **{**  **glutInit(&argc, argv); // Initialize GLUT**  **glutInitWindowSize(720, 720);**  **glutCreateWindow("TOWER"); // Create a window with the given title**  **// Set the window's initial width & height**  **glutDisplayFunc(display);**  **gluOrtho2D(-6,6,-6,4 );// Register display callback handler for window re-paint**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-**  **A drawing of a lighthouse  Description automatically generated** |

|  |
| --- |
| **Question- 3**  Design the given scenario  Traffic Light Drawing Vector Images (over 2,300) |
| **Graph Plot (Picture)-** |
| **Code-** |
| **Output Screenshot (Full Screen)-** |