**Lab Taks-4**

Submission Guidelines-

* Rename the file to your id only. If your id is 18-XXXXX-1, then the file name must be 18-XXXXX-1.docx.
* Must submit within time that will be discussed in class VUES to the section named Lab Tak-4
* Must include resources for all the section in the table

|  |
| --- |
| **Question- 1**  Draw the scenario of a traffic signal |
| **Graph Plot (Picture)-**  **(Not Needed)** |
| **Code-**  **#include <windows.h> // for MS Windows**  **#include <GL/glut.h> // GLUT, include glu.h and gl.h**  **#include<math.h>**  **/\* Initialize OpenGL Graphics \*/**  **void initGL() {**  **// Set "clearing" or background color**  **glClearColor(0.0f, 0.0f, 1.0f, 0.0f); // Black and opaque**  **}**  **void mountain()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.1f, 0.0f, 0.0f);**  **glShadeModel(GL\_SMOOTH);**  **glColor3f(0.0f,0.1f,0.0f);**  **glBegin(GL\_TRIANGLES);**  **glVertex2f(-1.1f, -0.5f);**  **glVertex2f(1.0f, -0.5f);**  **glVertex2f(0.5f, 0.2f);**  **glPopMatrix();**  **glEnd();**  **}**  **void tree()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(-0.5f, -0.1f, 0.0f);**  **glScalef(-0.3f, 0.9f, 0.0f);**  **glBegin(GL\_QUADS);**  **glColor3f(0.5f, 0.1f, 0.0f); //**  **glVertex2f(-0.1f, -0.9f); // x, y**  **glVertex2f(0.1f, -0.9f);**  **glVertex2f(0.1f, -0.3f); // x, y**  **glVertex2f(-0.1f, -0.3f);**  **glEnd();**  **glBegin(GL\_TRIANGLES);**  **glColor3f(1.0f, 1.0f, 0.0f); //**  **glVertex2f(-0.4f, -0.3f);**  **glVertex2f(0.4f, -0.3f);**  **glVertex2f(0.0f, 0.3f);**  **glEnd();**  **glBegin(GL\_TRIANGLES);**  **glColor3f(0.0f, 1.0f, 0.0f);**  **glVertex2f(-0.4f, 0.0f);**  **glVertex2f(0.4f, 0.0f);**  **glVertex2f(0.0f, 0.6f);**  **glEnd();**  **glPopMatrix();**  **}**  **void road()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f, 0.0f, 0.0f);**  **glColor3f(0.5f,0.5f,0.1f);**  **glBegin(GL\_QUADS);**  **glVertex2f(-1.0f, -1.0f);**  **glVertex2f(1.0f, -1.0f);**  **glVertex2f(1.0f, -0.5f);**  **glVertex2f(-1.1f, -0.5f);**  **glPopMatrix();**  **glEnd();**  **}**  **void lamp()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f, -0.1f, 0.0f);**  **glScalef(0.5f, 0.5f, 0.0f);**  **//glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer with current clearing color**  **glBegin(GL\_QUADS); // base**  **glColor3f(1.0f, 0.0f, 0.0f); // Red**  **glVertex2f(-0.3f, -0.9f); // x, y**  **glVertex2f(0.3f, -0.9f);**  **glVertex2f(0.3f, -0.8f); // x, y**  **glVertex2f(-0.3f, -0.8f);**  **glEnd();**  **//Scalef(0.6f, 1.5f, 0.0f);**  **glBegin(GL\_QUADS); // stand**  **glColor3f(1.0f, 0.0f, 0.0f); // Red**  **glVertex2f(-0.1f, -0.8f); // x, y**  **glVertex2f(0.1f, -0.8f);**  **glVertex2f(0.1f, -0.2f); // x, y**  **glVertex2f(-0.1f, -0.2f);**  **glEnd();**  **glScalef(0.5f, 2.0f, 0.0f);**  **glBegin(GL\_QUADS); //light**  **glColor3f(0.1f, 0.1f, 0.0f); // Red**  **glVertex2f(-0.20f,-0.2f); // x, y**  **glVertex2f(0.20, -0.2f);**  **glVertex2f(0.20, 0.1f); // x, y**  **glVertex2f(-0.20, 0.1f);**  **glEnd();**  **}**  **void lampLightYellow()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f,-0.15f,0.0f);**  **glColor3f(1.0f, 1.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(1.0f, 1.0f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.045;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void lampLightRed()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f,-0.05f,0.0f);**  **glColor3f(1.0f, 0.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(1.0f, 0.0f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.045;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void lampLightGreen()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f,-0.25f,0.0f);**  **glColor3f(0.0f, 0.5f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0.0f, 0.5f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.045;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void car()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(1.0f, -0.3f, 0.0f);**  **glScalef(1.3f, 0.6f, 0.0f);**  **glShadeModel (GL\_SMOOTH);**  **glColor3f(0.5f, 0.1f, 0.5f);**  **glBegin(GL\_POLYGON);**  **glVertex2f(-0.5f, -0.5f);**  **// x, y**  **glVertex2f(-0.1f, -0.5f);**  **glVertex2f(-0.1f, -0.2f);**  **glVertex2f(-0.2f, -0.1f);**  **glVertex2f(-0.2f, 0.0f);// x, y**  **glVertex2f(-0.5f, 0.0f);**  **glPopMatrix();**  **glEnd();**  **}**  **void carWheelOne()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.45f,-0.6f,0.0f);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0.0f, 0.0f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.085;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void carWheelTwo()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.75f,-0.6f,0.0f);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(0.0f, 0.0f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.085;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void sun()**  **{**  **glLoadIdentity(); //Reset the drawing perspective**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f,0.0f,0.0f);**  **glColor4f(1.0f, 1.0f, 0.0f, 0.0f);**  **glBegin(GL\_POLYGON);**  **for(int i=0;i<200;i++)**  **{**  **glColor4f(1.0f, 1.0f, 0.0f, 0.0f);**  **float pi=3.1416;**  **float A=(i\*2\*pi)/200;**  **float r=0.2;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x,y );**  **}**  **glEnd();**  **glPopMatrix();**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer with current clearing color**  **sun();**  **mountain();**  **tree();**  **road();**  **lamp();**  **lampLightRed();**  **lampLightYellow();**  **lampLightGreen();**  **car();**  **carWheelOne();**  **carWheelTwo();**  **glFlush(); // Render now**  **}**  **/\* Main function: GLUT runs as a console application starting at main() \*/**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitWindowSize(1320, 1320); // Initialize GLUT**  **glutCreateWindow("Vertex, Primitive & Color"); // Create window with the given title**  **gluOrtho2D(-10,10,-10,10); // Set the window's initial width & height**  **glutDisplayFunc(display); // Register callback handler for window re-paint event**  **initGL();**  **// Our own OpenGL initialization**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |