**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 <GL/glut.h>  #include<iostream>  #include <cmath>  float carX = 0.4;  float carY = -0.7;  float wheelAngle = 0.0;  float scaleValue = 1.0;  void Car()  {  glPushMatrix();  glTranslatef(carX, carY, 0.0);  glRotatef(wheelAngle, 0.0, 1.0, 0.0);  glScalef(scaleValue, scaleValue, 1.0);  glColor3f(0.0, 0.0, 1.0);  // Roof  glBegin(GL\_POLYGON);  glVertex2f(-0.4, 0.0);  glVertex2f(-0.3, 0.2);  glVertex2f(0.4, 0.2);  glVertex2f(0.4, 0.0);  glEnd();  // Body  glBegin(GL\_POLYGON);  glVertex2f(-0.4, -0.2);  glVertex2f(-0.4, 0.0);  glVertex2f(0.4, 0.0);  glVertex2f(0.4, -0.2);  glEnd();  // Windows  glColor3f(0.8, 0.8, 1.0);  glBegin(GL\_POLYGON);  glVertex2f(-0.30, 0.05);  glVertex2f(-0.30, 0.15);  glVertex2f(-0.10, 0.15);  glVertex2f(-0.10, 0.05);  glEnd();  glBegin(GL\_POLYGON);  glVertex2f(0.15, 0.05);  glVertex2f(0.15, 0.15);  glVertex2f(0.35, 0.15);  glVertex2f(0.35, 0.05);  glEnd();  //door  glBegin(GL\_LINES);  glColor3f(0.0, 0.0, 0.0);  glVertex2f(-0.05, -0.2);  glVertex2f(-0.05, 0.2);  glEnd();  // Wheels  glColor3f(0.0, 0.0, 0.0);  const float wheelRadius = 0.1;  const float wheelY = -0.2;  glBegin(GL\_POLYGON);  for (int i = 0; i < 360; i++)  {  float angle = i \* 3.14159 / 180.0;  float x = -0.25 + wheelRadius \* cos(angle);  float y = wheelY + wheelRadius \* sin(angle);  glVertex2f(x, y);  }  glEnd();  glBegin(GL\_POLYGON);  for (int i = 0; i < 360; i++)  {  float angle = i \* 3.14159 / 180.0;  float x = 0.25 + wheelRadius \* cos(angle);  float y = wheelY + wheelRadius \* sin(angle);  glVertex2f(x, y);  }  glEnd();  glPopMatrix();  }  void TrafficLight()  {  //pole  glColor3f(0.65, 0.65, 0.65);  glBegin(GL\_POLYGON);  glVertex2f(-0.95, -0.7);  glVertex2f(-0.95, 0.9);  glVertex2f(-0.9, 0.9);  glVertex2f(-0.9, -0.7);  glEnd();  glColor3f(0.5, 0.5, 0.5);  //light box  glBegin(GL\_POLYGON);  glVertex2f(-0.9, 0.6);  glVertex2f(-0.9, 0.9);  glVertex2f(-0.8, 0.9);  glVertex2f(-0.8, 0.6);  glEnd();  //red light  glColor3f(1.0, 0.0, 0.0);  glBegin(GL\_POLYGON);  for (int i = 0; i < 360; i++)  {  float angle = i \* 3.14159 / 180.0;  float x = -0.85 + 0.03 \* cos(angle);  float y = 0.85 + 0.03 \* sin(angle);  glVertex2f(x, y);  }  glEnd();  //yellow light  glColor3f(1.0, 1.0, 0.0);  glBegin(GL\_POLYGON);  for (int i = 0; i < 360; i++)  {  float angle = i \* 3.14159 / 180.0;  float x = -0.85 + 0.03 \* cos(angle);  float y = 0.75 + 0.03 \* sin(angle);  glVertex2f(x, y);  }  glEnd();  //green light  glColor3f(0.0, 1.0, 0.0);  glBegin(GL\_POLYGON);  for (int i = 0; i < 360; i++)  {  float angle = i \* 3.14159 / 180.0;  float x = -0.85 + 0.03 \* cos(angle);  float y = 0.65 + 0.03 \* sin(angle);  glVertex2f(x, y);  }  glEnd();  }  void Road()  {  glColor3f(86/255.0, 93/255.0, 109/255.0);  glBegin(GL\_POLYGON);  glVertex2f(-2.0, -0.7);  glVertex2f(-2.0, -1.25);  glVertex2f(2.0, -1.25);  glVertex2f(2.0, -0.7);  glEnd();  }  void Field()  {  glColor3f(104/255.0, 171/255.0, 58/255.0);  glBegin(GL\_POLYGON);  glVertex2f(-2.0, -1.25);  glVertex2f(-2.0, -1.7);  glVertex2f(2.0, -1.7);  glVertex2f(2.0, -1.25);  glEnd();  }  void Display()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  Field();  Road();  Car();  TrafficLight();  glFlush();  }  void Reshape(int w, int h)  {  glViewport(0, 0, w, h);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  glMatrixMode(GL\_MODELVIEW);  }  void Init()  {  glClearColor(116/255.0, 204/255.0, 243/255.0, 1.0);  }  int main(int argc, char\*\* argv)  {  glutInit(&argc, argv);  glutCreateWindow("Traffic Signal Scenario");  glutInitWindowSize(700, 400);  gluOrtho2D(-1.5, 1.5, -1.5, 1.5);  glutDisplayFunc(Display);  glutReshapeFunc(Reshape);  Init();  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-** |