STAMFORD UNIVERSITY BANGLADESH

Department of CSE

Batch:68(A), Semester: Summer 2022

Course Title: Computer Graphics Sessional

Course Code: CSI 414

Project Title:Traffic Signal System

Submitted To:

Saima Siddique Tashfia

Lecturer

Prepared By:

Rehana Kabir Mim **ID:CSE 068 07935**

Sheikh MD Shajahan Hosain Shium **ID:CSE 068 07948**

Date of Submission:18/11/2022

CODE:

```
#include<windows.h>
#include<GL/glut.h>
#include <GL/gl.h>
#include <stdlib.h>
#include<iostream>
using namespace std;
#define SPEED 35.0              //speed of traffic

float i=0.0;                    //movement of car

float m=0.0;                     //movement of clouds

float n=0.0;                    //movement of plane along x-axis
float o=0.0;                    //  and y-axis
float c=0.0;                    //movement of comet
float s=0.0;                              //movement of star
float a=0.0;
float b=0.0;
float r=0.0;
float t=0.0;
float q=0.0;
float z=0.0;
float j=0.0;
```

```
int light=1;                    //1 for green-light, 0 for red-light
int day=1;                      //1 for day ,0 for night
int plane=0;                    //1 for plane
int comet=0;
int star=1;                     //1 for comet
void draw_pixel(GLint x, GLint y)
{
        glBegin(GL_POINTS);
                glVertex2i(x,y);
        glEnd();
}
void plotpixels(GLint h,GLint u, GLint x,GLint y)
{
        draw_pixel(x+h,y+u);
        draw_pixel(-x+h,y+u);
        draw_pixel(x+h,-y+u);
        draw_pixel(-x+h,-y+u);
        draw_pixel(y+h,x+u);
        draw_pixel(-y+h,x+u);
        draw_pixel(y+h,-x+u);
        draw_pixel(-y+h,-x+u);
}
void draw_circle(GLint h, GLint u, GLint r)
{
```

```
        GLint p=1-r, x=0, y=r;//initialization
        while(y>x)
        {
                plotpixels(h,u,x,y);
                if(p<0) p=p+2*x+3;//updating value of p
                else
                {
                        p=p+2*(x-y)+5;
                        y=y-1;
                }
                x=x+1;
        }
        plotpixels(h,u,x,y);
}
void draw_object()
{
int l;
if(day==1)
{
                        //sky
glColor3f(0.0,0.9,0.9);
glBegin(GL_POLYGON);
   glVertex2f(0,450);
   glVertex2f(0,700);
```

```
   glVertex2f(1100,700);
   glVertex2f(1100,450);
glEnd();
                        //sun
        for(l=0;l<=35;l++)
{
                glColor3f(1.0,0.9,0.0);
                draw_circle(100,625,l);
}
                        //plane
if(plane==1)
{
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
        glVertex2f(925+n,625+o);
   glVertex2f(950+n,640+o);
        glVertex2f(1015+n,640+o);
        glVertex2f(1030+n,650+o);
        glVertex2f(1050+n,650+o);
        glVertex2f(1010+n,625+o);
glEnd();
glColor3f(0.8,0.8,0.8);
glBegin(GL_LINE_LOOP);
        glVertex2f(925+n,625+o);
```

```
   glVertex2f(950+n,640+o);
        glVertex2f(1015+n,640+o);
        glVertex2f(1030+n,650+o);
        glVertex2f(1050+n,650+o);
        glVertex2f(1010+n,625+o);
glEnd();
}                          //cloud1
        for(l=0;l<=20;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(160+m,625,l);
        }
        for(l=0;l<=35;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(200+m,625,l);
                draw_circle(225+m,625,l);
        }
        for(l=0;l<=20;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(265+m,625,l);
        }
                        //cloud2
```

```
        for(l=0;l<=20;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(370+m,615,l);
}
        for(l=0;l<=35;l++)
        {

                glColor3f(1.0,1.0,1.0);
                draw_circle(410+m,615,l);
                draw_circle(435+m,615,l);
                draw_circle(470+m,615,l);
        }

for(l=0;l<=20;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(500+m,615,l);
}
                        //grass
glColor3f(0.0,0.9,0.0);
glBegin(GL_POLYGON);
glVertex2f(0,160);
glVertex2f(0,450);
```

```
glVertex2f(1100,450);
glVertex2f(1100,160);
glEnd();

                    //pond
glColor3f(0.0,0.9,0.9);
glBegin(GL_POLYGON);
glVertex2f(25,350);
glVertex2f(25,375);
glVertex2f(50,400);
glVertex2f(75,410);
glVertex2f(100,420);
glVertex2f(200,420);
glVertex2f(225,410);
glVertex2f(250,405);
glVertex2f(275,390);
glVertex2f(300,375);
glVertex2f(310,350);
glVertex2f(300,320);
glVertex2f(275,300);
glVertex2f(250,295);
glVertex2f(225,290);
glVertex2f(200,285);
glVertex2f(175,280);
```

```
glVertex2f(150,280);
glVertex2f(125,280);
glVertex2f(100,290);
glVertex2f(75,300);
glVertex2f(50,310);
glEnd();
}
else
{                          //sky for night
glColor3f(0.0,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(0,450);
glVertex2f(0,700);
glVertex2f(1100,700);
glVertex2f(1100,450);
glEnd();
                                        //moon for night
int l;
        for(l=0;l<=35;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(100,625,l);
        }
                                    if(star==1)
```

```
{                                   //star1
glColor3f(1.0,1.0,1.0);
glBegin(GL_TRIANGLES);
glVertex2f(575+a,653+b);
glVertex2f(570+a,645+b);
glVertex2f(580+a,645+b);
glVertex2f(575+a,642+b);
glVertex2f(570+a,650+b);
glVertex2f(580+a,650+b);
glEnd();
}
else
{


}

if(star==1)
{
glColor3f(1.0,1.0,1.0);//star 2
glBegin(GL_TRIANGLES);
glVertex2f(975-r,643-t);
glVertex2f(970-r,635-t);
glVertex2f(980-r,635-t);
glVertex2f(975-r,632-t);
```

```
glVertex2f(970-r,640-t);
glVertex2f(980-r,640-t);
glEnd();
}
else
{

}
if(star==1)
{                          //star3
glColor3f(1.0,1.0,1.0);
glBegin(GL_TRIANGLES);

glVertex2f(870+q,535+z);
glVertex2f(880+q,535+z);
glVertex2f(875+q,532+z);
glVertex2f(870+q,540+z);
glVertex2f(880+q,540+z);
glEnd();
}
else
{

}
```

```
                    //star4
glColor3f(1.0,1.0,1.0);
glBegin(GL_TRIANGLES);
glVertex2f(375,598);
glVertex2f(370,590);
glVertex2f(380,590);
glVertex2f(375,587);
glVertex2f(370,595);
glVertex2f(380,595);
glEnd();




                    //star5
glColor3f(1.0,1.0,1.0);
glBegin(GL_TRIANGLES);
glVertex2f(750,628);
glVertex2f(745,620);
glVertex2f(755,620);
glVertex2f(750,618);
glVertex2f(745,625);
glVertex2f(755,625);
glEnd();
```

```
                    //star6
glColor3f(1.0,1.0,1.0);
glBegin(GL_TRIANGLES);
glVertex2f(200+q,628+z);
glVertex2f(195+q,620+z);
glVertex2f(205+q,620+z);
glVertex2f(200+q,618+z);
glVertex2f(195+q,625+z);
glVertex2f(205+q,625+z);

glEnd();



glColor3f(1.0,1.0,1.0);          //star7
glBegin(GL_TRIANGLES);
glVertex2f(500-a,543-b);
glVertex2f(495-a,535-b);
glVertex2f(505-a,535-b);
glVertex2f(500-a,532-b);
glVertex2f(495-a,540-b);
glVertex2f(505-a,540-b);
glEnd();     //star1
```

```
                        //comet for night
if(comet==1)
{
        for(l=0;l<=7;l++)
        {
                glColor3f(1.0,1.0,1.0);
                draw_circle(300+c,675,l);
        }
glColor3f(1.0,1.0,1.0);
        glBegin(GL_TRIANGLES);
        glVertex2f(200+c,675);
        glVertex2f(300+c,682);
        glVertex2f(300+c,668);
        glEnd();
}
                        //Plane for night
if(plane==1)
{
        for(l=0;l<=1;l++)
        {
                glColor3f(1.0,0.0,0.0);
                draw_circle(950+n,625+o,l);
                glColor3f(1.0,1.0,0.0);
```

```
            draw_circle(954+n,623+o,l);
        }
}
                        //grass for night
glColor3f(0.0,0.3,0.0);
glBegin(GL_POLYGON);
glVertex2f(0,160);
glVertex2f(0,450);
glVertex2f(1100,450);
glVertex2f(1100,160);
glEnd();
                        //pond for night
glColor3f(0.0,0.0,0.4);
glBegin(GL_POLYGON);
glVertex2f(25,350);
glVertex2f(25,375);
glVertex2f(50,400);
glVertex2f(75,410);
glVertex2f(100,420);
glVertex2f(200,420);
glVertex2f(225,410);
glVertex2f(250,405);
glVertex2f(275,390);
glVertex2f(300,375);
```

```
glVertex2f(310,350);
glVertex2f(300,320);
glVertex2f(275,300);
glVertex2f(250,295);
glVertex2f(225,290);
glVertex2f(200,285);
glVertex2f(175,280);
glVertex2f(150,280);
glVertex2f(125,280);
glVertex2f(100,290);
glVertex2f(75,300);
glVertex2f(50,310);
glEnd();
}
                    //road boundary
glColor3f(0.7,0.7,0.1);
glBegin(GL_POLYGON);
glVertex2f(0,150);
glVertex2f(0,160);




glColor3f(0.6,0.0,0.3);
glVertex2f(1100,160);
```

```
glVertex2f(1100,150);
glEnd();
                        //road
glColor3f(0.2,0.2,0.2);
glBegin(GL_POLYGON);
glVertex2f(0,0);
glVertex2f(0,150);
glVertex2f(1100,150);
glVertex2f(1100,0);
glEnd();
                        //tree
glColor3f(0.9,0.2,0.0);
glBegin(GL_POLYGON);
glVertex2f(350,325);
glVertex2f(350,395);
glVertex2f(365,395);
glVertex2f(365,325);
glEnd();
        for(l=0;l<=30;l++)
        {
                glColor3f(0.0,0.5,0.0);
                draw_circle(340,400,l);
                draw_circle(380,400,l);
        }
```

```
        for(l=0;l<=25;l++)
        {
                glColor3f(0.0,0.5,0.0);
                draw_circle(350,440,l);
                draw_circle(370,440,l);
        }
        for(l=0;l<=20;l++)
        {
                glColor3f(0.0,0.5,0.0);
                draw_circle(360,465,l);
        }
 glColor3f(0.9,0.9,0.9);//back compound
glBegin(GL_POLYGON);
glVertex2f(550,375);
glVertex2f(600,425);
glVertex2f(825,425);
glVertex2f(850,375);
glEnd();
```

//house

```
glColor3f(0.9,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(600,375);
glVertex2f(600,450);
glVertex2f(650,525);
glVertex2f(700,450);
glVertex2f(700,375);
glEnd();
                    //door
glColor3f(0.7,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(640,375);
glVertex2f(640,410);
glVertex2f(660,410);
glVertex2f(660,375);
glEnd();
                    //roof
glColor3f(0.5,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(700,450);
glVertex2f(650,525);
glVertex2f(750,525);
glVertex2f(780,450);
glEnd();
```

```
                    // window surface
glColor3f(0.8,0.8,0.2);
glBegin(GL_POLYGON);
glVertex2f(700,375);
glVertex2f(700,450);
glVertex2f(780,450);
glVertex2f(780,375);
glEnd();
                    //window
glColor3f(0.5,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(725,400);
glVertex2f(725,420);
glVertex2f(740,420);
glVertex2f(740,400);
glEnd();
glColor3f(0.7,0.7,0.7);//compound
glBegin(GL_POLYGON);
glVertex2f(550,325);
glVertex2f(550,375);
glVertex2f(850,375);
glVertex2f(850,325);
glEnd();
```

```
                    //signal
    glColor3f(1.0,0.0,0.0);
    glBegin(GL_POLYGON);
            glVertex2f(1060,160);
            glVertex2f(1060,350);
            glVertex2f(1070,350);
            glVertex2f(1070,160);
    glEnd();
    glColor3f(0.7,0.7,0.7);
    glBegin(GL_POLYGON);
            glVertex2f(1040,350);
            glVertex2f(1040,500);
            glVertex2f(1090,500);
            glVertex2f(1090,350);
    glEnd();
    for(l=0;l<=20;l++)
    {
   glColor3f(0.0,0.0,0.0);
            draw_circle(1065,475,l);

            glColor3f(0.0,0.0,0.0);
            draw_circle(1065,375,l);
    }
```

```
                    //car 1
glColor3f(0.9,0.2,0.0);
glBegin(GL_POLYGON);
glVertex2f(25+i,50);
glVertex2f(25+i,125);
glVertex2f(75+i,200);
glVertex2f(175+i,200);
glVertex2f(200+i,125);
glVertex2f(250+i,115);
glVertex2f(250+i,50);
glEnd();
                    //windows
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(35+i,125);
glVertex2f(80+i,190);
glVertex2f(115+i,190);
glVertex2f(115+i,125);
glEnd();
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(125+i,125);
glVertex2f(125+i,190);
glVertex2f(170+i,190);
```

```
glVertex2f(190+i,125);
glEnd();
for(l=0;l<20;l++)
  {
        glColor3f(0.0,0.0,0.0);
        draw_circle(75+i,50,l);
        draw_circle(175+i,50,l);
  }
                        //car2
glColor3f(0.3,0.0,0.5);
glBegin(GL_POLYGON);
glVertex2f(-470+i,50);
glVertex2f(-470+i,112);
glVertex2f(-400+i,125);
glVertex2f(-372+i,210);
glVertex2f(-210+i,210);
glVertex2f(-180+i,125);
glVertex2f(-135+i,125);
glVertex2f(-110+i,50);
glEnd();
                        //windows


 glColor3f(0.7,0.7,0.7);
```

```
glBegin(GL_POLYGON);
glVertex2f(550,325);
glVertex2f(550,375);
glVertex2f(850,375);
glVertex2f(850,325);
glEnd();



glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(-410+i,125);
glVertex2f(-364+i,200);
glVertex2f(-300+i,200);
glVertex2f(-300+i,125);
glEnd();
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
   glVertex2f(-290+i,125);
   glVertex2f(-290+i,200);
   glVertex2f(-217+i,200);
   glVertex2f(-192+i,125);
glEnd();
for(l=0;l<30;l++)
```

```
  {
        glColor3f(0.0,0.0,0.0);
        draw_circle(-350+i,50,l);
        draw_circle(-200+i,50,l);
  }
  //signal
  glColor3f(1.0,0.0,0.0);
        glBegin(GL_POLYGON);
              glVertex2f(1060,160);
              glVertex2f(1060,350);
              glVertex2f(1070,350);
              glVertex2f(1070,160);
        glEnd();
        glColor3f(0.7,0.7,0.7);
        glBegin(GL_POLYGON);
              glVertex2f(1040,350);
              glVertex2f(1040,500);
              glVertex2f(1090,500);
              glVertex2f(1090,350);
        glEnd();
        if(light==1)
{
      for(l=0;l<=20;l++)
        {
```

```
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,475,l);
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,425,l);
                glColor3f(0.2,1.0,0.0);
                draw_circle(1065,375,l);
        }
}
        else if(light==2)
{
  for(l=0;l<=20;l++)
        {
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,475,l);
                glColor3f(0.9,0.9,0.0);
                draw_circle(1065,425,l);
                glColor3f(0.0,0.0,0.0);
                draw_circle(1065,375,l);
        }
}
else
{
 for(l=0;l<=20;l++)
        {
```

```
            glColor3f(1.0,0.0,0.0);
            draw_circle(1065,475,l);
            glColor3f(0.0,0.0,0.0);
            draw_circle(1065,425,l);
            glColor3f(0.0,0.0,0.0);
            draw_circle(1065,375,l);
        }
}

            //code for bus
glColor3f(0.9,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(350+i,50);
glVertex2f(350+i,275);
glVertex2f(722+i,275);
glVertex2f(750+i,175);
glVertex2f(750+i,50);
glEnd();
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
glVertex2f(650+i,175);
glVertex2f(650+i,260);
glVertex2f(720+i,260);
glVertex2f(745+i,175);
```

```
glEnd();
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
glVertex2f(550+i,175);
glVertex2f(550+i,260);
glVertex2f(625+i,260);
glVertex2f(625+i,175);
glEnd();
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
glVertex2f(450+i,175);
glVertex2f(450+i,260);
glVertex2f(525+i,260);
glVertex2f(525+i,175);
glEnd();
glColor3f(1.0,1.0,1.0);
glBegin(GL_POLYGON);
glVertex2f(375+i,175);
glVertex2f(375+i,260);
glVertex2f(425+i,260);
glVertex2f(425+i,175);
glEnd();
for(l=0;l<30;l++)
  {
```

```
        glColor3f(0.0,0.0,0.0);
        draw_circle(450+i,50,l);
        draw_circle(625+i,50,l);
  }
glFlush();
}

void idle()
{
glClearColor(1.0,1.0,1.0,1.0);
if(light==0 && (i>=330 && i<=750))  //value of i when first vehicle is near the traffic-signal
 {
         i+=SPEED/10;
    ++m;
        n-=2;
         o+=0.2;
        c+=2;
        j-=3;
   a-=2;
        b-=0.2;

        r-=4;
        t-=0.1;
```

```
   q-=5;
   z-=0.8;
 }
if(light==0 && (i>=830 && i<=1100)) //value of i when second vehicle
is near the traffic-signal
 {
        i+=SPEED/10;
    ++m;
       n-=2;
        o+=0.2;
       c+=2;
       j-=3;
   a-=2;
       b-=0.2;

       r-=4;
       t-=0.1;
   q-=5;
   z-=0.8;
 }
if(light==0 && (i>=1200 && i<=1620))// value of i when third vehicle
is near the traffic signal
 {
        i+=SPEED/10;
```

```
    ++m;
        n-=2;
         o+=0.2;
        c+=2;
   j-=3;
   a-=2;
        b-=0.2;


        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
 }
else if(light==2 && (i>=330 && i<=750))  //value of i when first
vehicle is near the traffic-signal
 {
         i+=SPEED/20;
    ++m;
        n-=2;
         o+=0.2;
        c+=2;
   j-=3;
   a-=2;
        b-=0.2;
```

```
        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
 }
else if(light==2 && (i>=830 && i<=1100)) //value of i when second vehicle is near the traffic-signal
 {
         i+=SPEED/20;
    ++m;
        n-=2;
         o+=0.2;
        c+=2;
   j-=3;
   a-=2;
        b-=0.2;

        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
 }
```

```
else if(light==2 && (i>=1200 && i<=1620))// value of i when third vehicle is near the traffic signal
 {
        i+=SPEED/20;
    ++m;
       n-=2;
        o+=0.2;
       c+=2;
   j-=3;
   a-=2;
       b-=0.2;

       r-=4;
       t-=0.1;
   q-=5;
   z-=0.8;
 }
if(light==0)
 {
        i=i;
        ++m;
       n-=2;
        o+=0.2;
       c+=2;
```

```
   j-=3;
   a-=2;
        b-=0.2;

        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
 }
else if(light==2)
 {
         i=i;
         ++m;
        n-=2;
         o+=0.2;
        c+=2;
   j-=3;
   a-=2;
        b-=0.2;

        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
```

```
 }
else
 {
    i+=SPEED/10;
    ++m;
        n-=2;
         o+=0.2;
        c+=2;
   j-=3;
   a-=2;
        b-=0.2;

        r-=4;
        t-=0.1;
   q-=5;
   z-=0.8;
 }
if(i>1630)
         i=0.0;
if(m>1100)
         m=0.0;
if( o>75)
 {
        plane=0;
```

```
 }
if(c>500)
 {
        comet=0;
 }
glutPostRedisplay();
}
void keyboardFunc( unsigned char key, int x, int y )
{
switch( key )
   {
case 'g':
case 'G':
light=1;
break;
case 'y':
case 'Y':
light=2;
break;
        case 'r':
        case 'R':
                light=0;
                break;
case 'd':
```

```
        case 'D':
                day=1;
                break;

        case 'n':
case 'N':
                day=0;
                break;

                case 's':
case 'S':
                star=1;
                a=b=0.0;
                break;
   };

}
void main_menu(int index)
{
        switch(index)
        {
        case 1:
        if(index==1)
         {
```

```
        plane=1;
                o=n=0.0;
         }
        break;
        case 2:
        if(index==2)
         {
                comet=1;
                 c=0.0;
         }
        break;
        case 3:
        if(index==3)
         {
                exit(0);
         }
        break;
        }
}
void myinit()
{
glClearColor(1.0,1.0,1.0,1.0);
glColor3f(0.0,0.0,1.0);
glPointSize(2.0);
```

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0,1100.0,0.0,700.0);
}
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
draw_object();

glFlush();
}
int main(int argc,char*argv[])
{
int c_menu;
   cout<<("\n");
   cout<<(" #----Graphics Project:-'Simulation of Traffic Signal
System'------#\n");
     cout<<(" |-----------------------------------------------------|\n");
    cout<<(" |                                                     |\n");
   cout<<(" #----Help Center (How to Operate ?) ----------------------
#\n");
         cout<<(" |    |> Press 'r' or 'R' to change the signal light to Red
|\n");
         cout<<(" |    |> Press 'g' or 'Y' to change the signal light to
Yellow  |\n");
```

```
        cout<<(" |    |> Press 'g' or 'G' to change the signal light to
Green|\n");
        cout<<(" |    |> Press 'd' or 'D' to make it Day Time
|\n");
        cout<<(" |    |> Press 'n' or 'N' to make it Night Time
|\n");
        cout<<(" |    |> Press 's' or 'S' to make fallen star
|\n");
        cout<<(" |    |> Press RIGHT MOUSE BUTTON to display menu
|\n");
        cout<<(" |      |> Select  'Aeroplane' to add moving Aeroplane
|\n");
       cout<<(" |      |> Select  'Comet' to add moving Comet
|\n");
        cout<<(" |      |> Select  'Quite' to Exit the application
|\n");
        cout<<(" |                                                |\n");
        cout<<("
|================================================================
====|\n");
        cout<<(" |--------------Special thanks to SULTAN AL GAIB----------
---|\n");
        cout<<("
|================================================================
====|\n");
    glutInit(&argc, argv);
```

```
	glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
	glutInitWindowSize(1100.0,700.0);
	glutInitWindowPosition(250,0);
	glutCreateWindow("Traffic Signal System");
	myinit();
	glutDisplayFunc(display);
	glutIdleFunc(idle);
	glutKeyboardFunc(keyboardFunc);
	c_menu=glutCreateMenu(main_menu);
	glutAddMenuEntry("Aeroplane",1);
	glutAddMenuEntry("Comet",2);
	glutAddMenuEntry("Quite",3);
	glutAttachMenu(GLUT_RIGHT_BUTTON);
	glutMainLoop();
	return 0;
}
```