

computer graphics lab

Cse422



August 27, 2021

Submitted To

Hasin Rehana

Lecturer of computer graphics lab

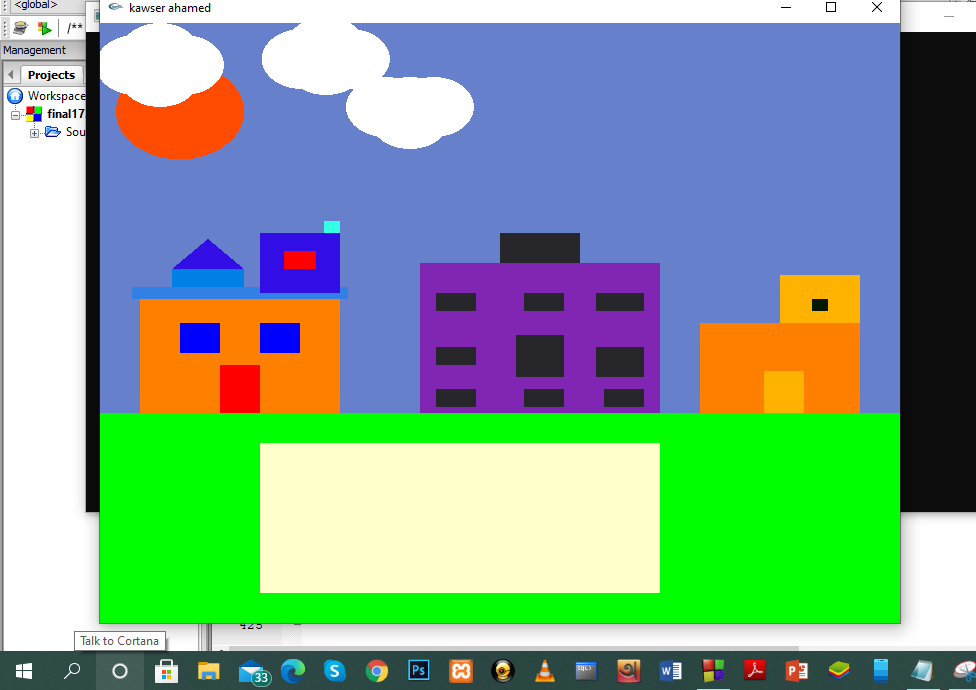
Submitted by

Md,Kawser Ahamed

Id:181-15-1722

Section :pc-f

Output:



Code:

#include<windows.h>

#ifdef \_APPLE\_

#else

#include<GL/glut.h>

#endif

#include<stdlib.h>

#include<stdio.h>

#include<math.h>

float theta;

float radius=0.08;

float r=0.05;

float r1=0.1;

float position;

float p=-20.0;

float c=200.0;

float c2=230;

float cl1=-0.30;

float cl2=-0.20;

float cl3=-0.40;

float x=0;

float sun=0;

void display(void)

{

glClear (GL\_COLOR\_BUFFER\_BIT);

//sky

glBegin(GL\_QUADS);

glColor3f (0.4, 0.5, 0.8);

glVertex3f(0.0f, 0.35f, 0.0f);

glVertex3f(1.0f, 0.35f, 0.0f);

glVertex3f(1.0f, 1.0f, 0.0f);

glVertex3f(0.0f, 1.0f, 0.0f);

glEnd();

//sun

if(sun<=0.1)

sun=sun-0.00001;

else

sun=0;

glutPostRedisplay();

float prevX = 0, prevY = 0;

glColor3f(1.0, 0.3, 0.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(0.1+radius\*sin(theta),sun+0.9-radius\*cos(theta));

}

glEnd();

//cloud 1

if(cl1<=1.0)

cl1=cl1+.00002;

else

cl1=-0.1;

glutPostRedisplay();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl1+0.31+r\*sin(theta),0.93-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl1+0.25+r\*sin(theta),0.93-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl1+0.28+r\*sin(theta),0.95-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl1+0.28+r\*sin(theta),0.91-r\*cos(theta));

}

glEnd();

//cloud 2

if(cl2<=0.80)

cl2=cl2+.00001;

else

cl2=-0.20;

glutPostRedisplay();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl2+0.57+r\*sin(theta),0.86-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl2+0.51+r\*sin(theta),0.86-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl2+0.54+r\*sin(theta),0.86-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl2+0.54+r\*sin(theta),0.84-r\*cos(theta));

}

glEnd();

//cloud 3

if(cl3<=1.0)

cl3=cl3+.00003;

else

cl3=-0.40;

glutPostRedisplay();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl3+0.57+r\*sin(theta),0.94-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl3+0.51+r\*sin(theta),0.94-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl3+0.54+r\*sin(theta),0.96-r\*cos(theta));

}

glEnd();

glColor3f(1.0, 1.0, 1.0);

glBegin(GL\_POLYGON);

for(int i=0; i<360; i++)

{

theta=i\*3.142/180;

glVertex2f(cl3+0.54+r\*sin(theta),0.93-r\*cos(theta));

}

glEnd();

//house1

glBegin(GL\_QUADS);

glColor3f (1.0, 0.5, 0.0);

glVertex3f(0.05f, 0.35f, 0.0f);

glVertex3f(0.30f, 0.35f, 0.0f);

glVertex3f(0.30f, 0.54f, 0.0f);

glVertex3f(0.05f, 0.54f, 0.0f);

//door

glColor3f (1.0, 0.0, 0.0);

glVertex3f(0.15f, 0.35f, 0.0f);

glVertex3f(0.20f, 0.35f, 0.0f);

glVertex3f(0.20f, 0.43f, 0.0f);

glVertex3f(0.15f, 0.43f, 0.0f);

//left side window

glColor3f (0.0, 0.0, 1.0);

glVertex3f(0.10f, 0.45f, 0.0f);

glVertex3f(0.15f, 0.45f, 0.0f);

glVertex3f(0.15f, 0.50f, 0.0f);

glVertex3f(0.10f, 0.50f, 0.0f);

//right side window

glColor3f (0.0, 0.0, 1.0);

glVertex3f(0.20f, 0.45f, 0.0f);

glVertex3f(0.25f, 0.45f, 0.0f);

glVertex3f(0.25f, 0.50f, 0.0f);

glVertex3f(0.20f, 0.50f, 0.0f);

//border

glColor3f (0.2, .50, 0.9);

glVertex3f(0.04f, 0.54f, 0.0f);

glVertex3f(0.31f, 0.54f, 0.0f);

glVertex3f(0.31f, 0.56f, 0.0f);

glVertex3f(0.04f, 0.56f, 0.0f);

glColor3f (0.0, 0.5, 0.9);

glVertex3f(0.09f, 0.56f, 0.0f);

glVertex3f(0.18f, 0.56f, 0.0f);

glVertex3f(0.18f, 0.59f, 0.0f);

glVertex3f(0.09f, 0.59f, 0.0f);

//right side room

glColor3f (0.20, .05, 0.9);

glVertex3f(0.20f, 0.55f, 0.0f);

glVertex3f(0.30f, 0.55f, 0.0f);

glVertex3f(0.30f, 0.65f, 0.0f);

glVertex3f(0.20f, 0.65f, 0.0f);

//window

glColor3f (1.0, 0.0, 0.0);

glVertex3f(0.23f, 0.59f, 0.0f);

glVertex3f(0.27f, 0.59f, 0.0f);

glVertex3f(0.27f, 0.62f, 0.0f);

glVertex3f(0.23f, 0.62f, 0.0f);

glColor3f (0.2, 1.0, 0.9);

glVertex3f(0.28f, 0.65f, 0.0f);

glVertex3f(0.30f, 0.65f, 0.0f);

glVertex3f(0.30f, 0.67f, 0.0f);

glVertex3f(0.28f, 0.67f, 0.0f);

glEnd();

//house left room

glBegin(GL\_TRIANGLES);

glColor3f (0.2, .05, 0.9);

glVertex3f(0.09f, 0.59f, 0.0f);

glVertex3f(0.18f, 0.59f, 0.0f);

glVertex3f(0.135f, 0.64f, 0.0f);

glEnd();

//house2

glBegin(GL\_QUADS);

glColor3f (0.5, 0.15, 0.7);

glVertex3f(0.40f, 0.35f, 0.0f);

glVertex3f(0.70f, 0.35f, 0.0f);

glVertex3f(0.70f, 0.60f, 0.0f);

glVertex3f(0.40f, 0.60f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.42f, 0.36f, 0.0f);

glVertex3f(0.47f, 0.36f, 0.0f);

glVertex3f(0.47f, 0.39f, 0.0f);

glVertex3f(0.42f, 0.39f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.53f, 0.36f, 0.0f);

glVertex3f(0.58f, 0.36f, 0.0f);

glVertex3f(0.58f, 0.39f, 0.0f);

glVertex3f(0.53f, 0.39f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.63f, 0.36f, 0.0f);

glVertex3f(0.68f, 0.36f, 0.0f);

glVertex3f(0.68f, 0.39f, 0.0f);

glVertex3f(0.63f, 0.39f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.42f, 0.43f, 0.0f);

glVertex3f(0.47f, 0.43f, 0.0f);

glVertex3f(0.47f, 0.46f, 0.0f);

glVertex3f(0.42f, 0.46f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.52f, 0.41f, 0.0f);

glVertex3f(0.58f, 0.41f, 0.0f);

glVertex3f(0.58f, 0.48f, 0.0f);

glVertex3f(0.52f, 0.48f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.62f, 0.41f, 0.0f);

glVertex3f(0.68f, 0.41f, 0.0f);

glVertex3f(0.68f, 0.46f, 0.0f);

glVertex3f(0.62f, 0.46f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.42f, 0.52f, 0.0f);

glVertex3f(0.47f, 0.52f, 0.0f);

glVertex3f(0.47f, 0.55f, 0.0f);

glVertex3f(0.42f, 0.55f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.53f, 0.52f, 0.0f);

glVertex3f(0.58f, 0.52f, 0.0f);

glVertex3f(0.58f, 0.55f, 0.0f);

glVertex3f(0.53f, 0.55f, 0.0f);

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.62f, 0.52f, 0.0f);

glVertex3f(0.68f, 0.52f, 0.0f);

glVertex3f(0.68f, 0.55f, 0.0f);

glVertex3f(0.62f, 0.55f, 0.0f);

//house2 roof

glColor3f (0.15, 0.15, 0.17);

glVertex3f(0.50f, 0.60f, 0.0f);

glVertex3f(0.60f, 0.60f, 0.0f);

glVertex3f(0.60f, 0.65f, 0.0f);

glVertex3f(0.50f, 0.65f, 0.0f);

//houde-3

glColor3f (1.0, 0.5, 0.0);

glVertex3f(0.75f, 0.35f, 0.0f);

glVertex3f(0.95f, 0.35f, 0.0f);

glVertex3f(0.95f, 0.50f, 0.0f);

glVertex3f(0.75f, 0.50f, 0.0f);

glColor3f (1.0, 0.7, 0.0);

glVertex3f(0.83f, 0.35f, 0.0f);

glVertex3f(0.88f, 0.35f, 0.0f);

glVertex3f(0.88f, 0.42f, 0.0f);

glVertex3f(0.83f, 0.42f, 0.0f);

//house3 roof

glColor3f (1.0, 0.7, 0.0);

glVertex3f(0.85f, 0.50f, 0.0f);

glVertex3f(0.95f, 0.50f, 0.0f);

glVertex3f(0.95f, 0.58f, 0.0f);

glVertex3f(0.85f, 0.58f, 0.0f);

//window

glColor3f (0.0, 0.1, 0.0);

glVertex3f(0.89f, 0.52f, 0.0f);

glVertex3f(0.91f, 0.52f, 0.0f);

glVertex3f(0.91f, 0.54f, 0.0f);

glVertex3f(0.89f, 0.54f, 0.0f);

glEnd();

//ground

glBegin(GL\_QUADS);

glColor3f (0.0, 1.0, 0.0);

glVertex3f(0.0f, 0.0f, 0.0f);

glVertex3f(1.0f, 0.0f, 0.0f);

glVertex3f(1.0f, 0.35f, 0.0f);

glVertex3f(0.0f, 0.35f, 0.0f);

//play ground

glColor3f (1.0, 1.0, 0.8);

glVertex3f(0.20f, 0.05f, 0.0f);

glVertex3f(0.70f, 0.05f, 0.0f);

glVertex3f(0.70f, 0.30f, 0.0f);

glVertex3f(0.20f, 0.30f, 0.0f);

glEnd();

glFlush ();

}

void init (void)

{

/\* select clearing (background) color \*/

glClearColor (1.0, 1.0, 1.0, 0.0);

/\* initialize viewing values \*/

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(0.0, 1.0, 0.0, 1.0, -10.0, 10.0);

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode (GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize (800, 600);

glutInitWindowPosition (100, 100);

glutCreateWindow ("kawser ahamed ");

init ();

glutDisplayFunc(display);

glutMainLoop();

return 0; /\* ISO C requires main to return int. \*/

}