**Program 6-**

**To draw a simple shaded scene consisting of a tea pot on a table. Define suitably the position and properties of the light source along with the properties of the surfaces of the solid object used in the scene.**

**Program-**

#include<stdio.h>

#include<GL/glut.h>

void init()

{

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-4,4,-4,4,-10,10);

glMatrixMode(GL\_MODELVIEW);

}

void wall()

{

glPushMatrix();

glScalef(2,0.05,2);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(-2,2,0);

glScalef(0.05,2,2);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(0,2,-2);

glScalef(2,2,0.05);

glutSolidCube(2);

glPopMatrix();

}

void table()

{

glPushMatrix();

glTranslatef(0,0.5,0);

glScalef(1,0.05,1);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(-0.8,0.2,0.8);

glScalef(0.1,0.25,0.1);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(0.8,0.2,0.8);

glScalef(0.1,0.25,0.1);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(0.8,0.2,-0.8);

glScalef(0.1,0.25,0.1);

glutSolidCube(2);

glPopMatrix();

glPushMatrix();

glTranslatef(-0.8,0.2,-0.8);

glScalef(0.1,0.25,0.1);

glutSolidCube(2);

glPopMatrix();

}

void tea()

{

glPushMatrix();

glTranslatef(0,1,0);

glutSolidTeapot(0.5);

glPopMatrix();

}

void display(void)

{

float amb[]={1,0,0};

float pos[]={2,4,1};

glClear(GL\_COLOR\_BUFFER\_BIT|GL\_DEPTH\_BUFFER\_BIT);

glMaterialfv(GL\_FRONT,GL\_AMBIENT,amb);

glLightfv(GL\_LIGHT0,GL\_POSITION,pos);

gluLookAt(2,1,2,0,1,0,0,1,0);

wall();

table();

tea();

glFlush();

}

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

{

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE|GLUT\_DEPTH);

glutInitWindowSize(1200,1200);

glutCreateWindow("Teapot");

init();

glutDisplayFunc(display);

glEnable(GL\_DEPTH\_TEST);

glShadeModel(GL\_SMOOTH);

glEnable(GL\_LIGHTING);

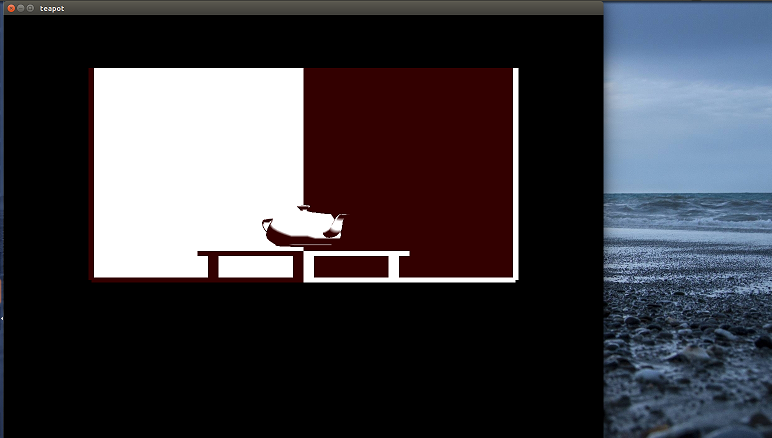
glEnable(GL\_LIGHT0);

glEnable(GL\_NORMALIZE);

glutMainLoop();

}

**Output:**



**Program 8-**

**Develop a menu driven program to animate a flag using Bezier Curve algorithm**

#include<GL/glut.h>

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#define PI 3.1416

static int win,val=0,CMenu;

void CreateMenu(void);

void Menu(int value);

struct wcPt3D

{

GLfloat x, y, z;

};

GLsizei winWidth = 600, winHeight = 600;

GLfloat xwcMin = 0.0, xwcMax = 130.0;

GLfloat ywcMin = 0.0, ywcMax = 130.0;

void bino(GLint n, GLint \*C)

{

GLint k, j;

for(k=0;k<=n;k++)

{

C[k]=1;

for(j=n;j>=k+1; j--)

C[k]\*=j;

for(j=n-k;j>=2;j--)

C[k]/=j;

}

}

void computeBezPt(GLfloat u,struct wcPt3D \*bezPt, GLint nCtrlPts,struct wcPt3D \*ctrlPts, GLint \*C)

{

GLint k, n=nCtrlPts-1;

GLfloat bezBlendFcn;

bezPt ->x =bezPt ->y = bezPt->z=0.0;

for(k=0; k< nCtrlPts; k++)

{

bezBlendFcn = C[k] \* pow(u, k) \* pow( 1-u, n-k);

bezPt ->x += ctrlPts[k].x \* bezBlendFcn;

bezPt ->y += ctrlPts[k].y \* bezBlendFcn;

bezPt ->z += ctrlPts[k].z \* bezBlendFcn;

}

}

void bezier(struct wcPt3D \*ctrlPts, GLint nCtrlPts, GLint nBezCurvePts)

{

struct wcPt3D bezCurvePt;

GLfloat u;

GLint \*C, k;

C= new GLint[nCtrlPts];

bino(nCtrlPts-1, C);

glBegin(GL\_LINE\_STRIP);

for(k=0; k<=nBezCurvePts; k++)

{

u=GLfloat(k)/GLfloat(nBezCurvePts);

computeBezPt(u, &bezCurvePt, nCtrlPts, ctrlPts, C);

glVertex2f(bezCurvePt.x, bezCurvePt.y);

}

glEnd();

delete[]C;

}

void displayFcn()

{

GLint nCtrlPts = 4, nBezCurvePts =20;

static float theta = 0;

struct wcPt3D ctrlPts[4] = {{20, 100, 0},{30, 110, 0},{50, 90, 0},{60, 100, 0}};

ctrlPts[1].x +=10\*sin(theta \* PI/180.0);

ctrlPts[1].y +=5\*sin(theta \* PI/180.0);

ctrlPts[2].x -= 10\*sin((theta+30) \* PI/180.0);

ctrlPts[2].y -= 10\*sin((theta+30) \* PI/180.0);

ctrlPts[3].x-= 4\*sin((theta) \* PI/180.0);

ctrlPts[3].y += sin((theta-30) \* PI/180.0);

theta+=0.1;

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(1.0, 1.0, 1.0);

glPointSize(5);

//Indian Flag

if(val==1){

glPushMatrix();

glLineWidth(5);

glColor3f(1.0,0.5,0); //Indian flag: Orange color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glColor3f(1,1,1); //Indian flag: white color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glColor3f(0,1.0,0); //Indian flag: green color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glPopMatrix();

glColor3f(0.7, 0.5,0.3);

glLineWidth(5);

glBegin(GL\_LINES);

glVertex2f(20,100);

glVertex2f(20,40);

glEnd();

glFlush();

}

//Karnataka Flag

if(val==2){

glPushMatrix();

glLineWidth(5);

glColor3f(1.0, 1.0, 0.0); //Karnataka flag: Yellow color code

for(int i=0;i<12;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glColor3f(1, 0.0, 0.0); //Karnataka flag: Red color code

for(int i=0;i<12;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glPopMatrix();

glColor3f(0.7, 0.5,0.3);

glLineWidth(5);

glBegin(GL\_LINES);

glVertex2f(20,100);

glVertex2f(20,40);

glEnd();

glFlush();

}

glutPostRedisplay();

glutSwapBuffers();

}

void winReshapeFun(GLint newWidth, GLint newHeight)

{

glViewport(0, 0, newWidth, newHeight);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(xwcMin, xwcMax, ywcMin, ywcMax);

glClear(GL\_COLOR\_BUFFER\_BIT);

}

void CreateMenu(void)

{

CMenu= glutCreateMenu(Menu);//Creaate Menu Option

glutAddMenuEntry("Indian Flag",1);

glutAddMenuEntry("Karnataka Flag",2);

glutAddMenuEntry("Exit",0);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

}

void Menu(int value)

{

if(value==0)

{

glutDestroyWindow(win);

exit(0);

}

else {

val=value;

}

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowPosition(50, 50);

glutInitWindowSize(winWidth, winHeight);

glutCreateWindow("Prg. 8 Bezier Curve");

CreateMenu();

glutDisplayFunc(displayFcn);

glutReshapeFunc(winReshapeFun);

glutMainLoop();

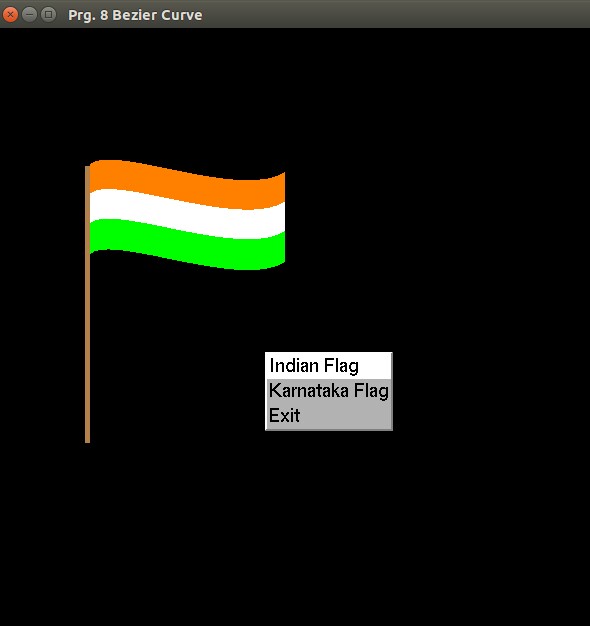
}

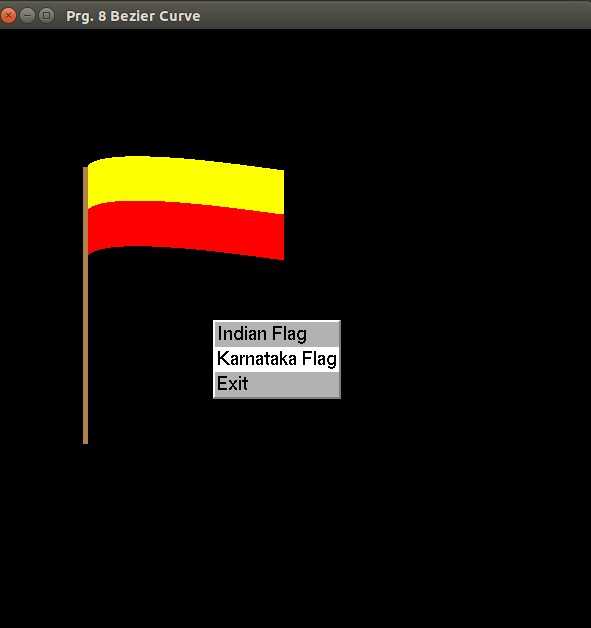
**NOTE:**

This is a c++ program. We nedd to save this program file name with .cpp extension and for compilation we need to use following commands.

g++ filename.cpp –lglut lGLU –lGL -lm

**Output:**





**Program 8-**

**Develop a menu driven program to fill the polygon using scan line algorithm.**

#include<stdlib.h>

#include<stdio.h>

#include<GL/glut.h>

float x1,x2,x3,x4,y1,y2,y3,y4;

static int win,val=0,CMenu;

void CreateMenu(void);

void Menu(int value);

void edgedetect(float x1,float y1,float x2,float y2,int \*le,int \*re)

{

float mx,x,temp;

int i;

if((y2-y1)<0)

{

temp=y1;y1=y2;y2=temp;

temp=x1;x1=x2;x2=temp;

}

if((y2-y1)!=0)

mx=(x2-x1)/(y2-y1);

else

mx=x2-x1;

x=x1;

for(i=y1;i<=y2;i++)

{

if(x<(float)le[i])

le[i]=(int)x;

if(x>(float)re[i])

re[i]=(int)x;

x+=mx;

}

}

void draw\_pixel(int x,int y)

{

if(val==1)

{

glColor3f(1.0,0.0,0.0);

}

else if(val==2)

{

glColor3f(0.0,0.0,1.0);

}

else if(val==3)

{

glColor3f(1.0,0.5,0.0);

}

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

void scanfill(float x1,float y1,float x2,float y2,float x3,float y3,float x4,float y4)

{

int le[500],re[500];

int i,y;

for(i=0;i<500;i++)

{

le[i]=500;

re[i]=0;

}

edgedetect(x1,y1,x2,y2,le,re);

edgedetect(x2,y2,x3,y3,le,re);

edgedetect(x3,y3,x4,y4,le,re);

edgedetect(x4,y4,x1,y1,le,re);

for(y=0;y<500;y++)

{

if(le[y]<=re[y])

for(i=(int)le[y];i<(int)re[y];i++)

draw\_pixel(i,y);

}

}

void display()

{

x1=200.0;y1=200.0;x2=100.0;y2=300.0;x3=200.0;

y3=400.0;x4=300.0;y4=300.0;

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(0.0,1.0,0.0);

glBegin(GL\_LINE\_LOOP);

glVertex2f(x1,y1);

glVertex2f(x2,y2);

glVertex2f(x3,y3);

glVertex2f(x4,y4);

glEnd();

scanfill(x1,y1,x2,y2,x3,y3,x4,y4);

glFlush();

}

void myinit()

{

glClearColor(1.0,1.0,1.0,1.0);

glColor3f(1.0,0.0,0.0);

glPointSize(1.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0.0,499.0,0.0,499.0);

}

void CreateMenu(void)

{

CMenu= glutCreateMenu(Menu);//Creaate Menu Option

glutAddMenuEntry("Red",1);

glutAddMenuEntry("Blue",2);

glutAddMenuEntry("Orange",3);

glutAddMenuEntry("Exit",0);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

}

void Menu(int value)

{

if(value==0)

{

glutDestroyWindow(win);

exit(0);

}

else {

val=value;

}

}

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

{

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500,500);

glutInitWindowPosition(0,0);

win=glutCreateWindow("Prg 9. Scan line Polygon filling algorithm");

CreateMenu();

glutDisplayFunc(display);

myinit();

glutMainLoop();

}

**Output:**

