1. Develop a menu driven program to animate a flag using Bezier curve algorithm.

```
#include<GL/glut.h>
#include<stdio.h>
#include<math.h>
#define PI 3.1416
typedef struct point
{
  GLfloat x, y, z;
};
void bino(int n, int *C)
{
 int 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(float u, point *pt1, int cPt, point *pt2, int *C)
{
  int k, n=cPt-1;
  float bFcn;
  pt1 ->x =pt1 ->y = pt1->z=0.0;
  for(k=0; k< cPt; k++)
  {
    bFcn = C[k] * pow(u, k) * pow( 1-u, n-k);
    pt1 ->x += pt2[k].x * bFcn;
    pt1 ->y += pt2[k].y * bFcn;
    pt1 ->z += pt2[k].z * bFcn;
  }
}
void bezier(point *pt1, int cPt, int bPt)
{
  point bcPt;
  float u;
  int *C, k;
  C= new int[cPt];
  bino(cPt-1, C);
  glBegin(GL_LINE_STRIP);
  for(k=0; k<=bPt; k++)
```

```
 {
  u=float(k)/float(bPt);
  computeBezPt(u, &bcPt, cPt, pt1, C);
  glVertex2f(bcPt.x, bcPt.y);
 }
 glEnd();
 delete[]C;
}
```

```
float theta = 0;
void display()
{
glClear(GL_COLOR_BUFFER_BIT);
 int nCtrlPts = 4, nBCPts =20;
 point ctrlPts[4] = {{100, 400, 0}, {150, 450, 0}, {250, 350, 0},
 {300, 400, 0}};
 ctrlPts[1].x +=50*sin(theta * PI/180.0);
 ctrlPts[1].y +=25*sin(theta * PI/180.0);
 ctrlPts[2].x -= 50*sin((theta+30) * PI/180.0);
 ctrlPts[2].y -= 50*sin((theta+30) * PI/180.0);
 ctrlPts[3].x -= 25*sin((theta) * PI/180.0);
 ctrlPts[3].y += sin((theta-30) * PI/180.0);
 theta+=0.2;
 glClear(GL_COLOR_BUFFER_BIT);
 glColor3f(1.0, 1.0, 1.0);
 glPointSize(5);
 glPushMatrix();
 glLineWidth(5);
 glColor3f(1, 0.4, 0.2); //Indian flag: Orange color code
 for(int i=0;i<50;i++)
 {
  glTranslatef(0, -0.8, 0);
  bezier(ctrlPts, nCtrlPts, nBCPts);
 }
 glColor3f(1, 1, 1); //Indian flag: white color code
 for(int i=0;i<50;i++)
 {
  glTranslatef(0, -0.8, 0);
  bezier(ctrlPts, nCtrlPts, nBCPts);
 }
 glColor3f(0, 1, 0); //Indian flag: green color code
```

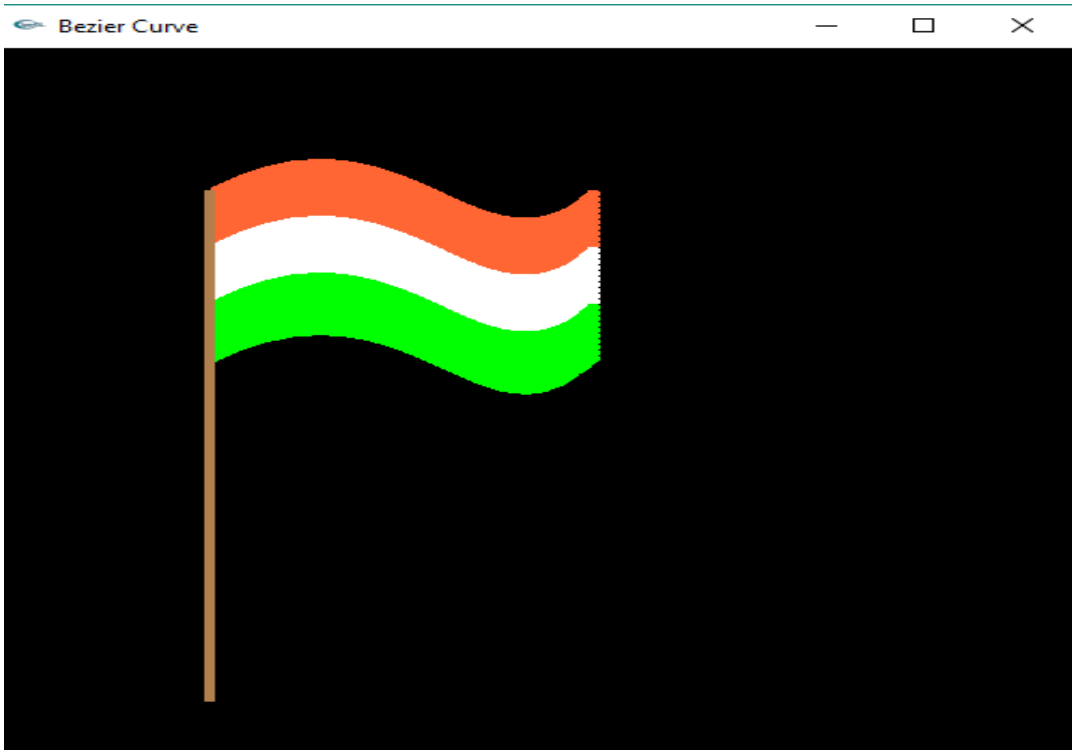
```
 for(int i=0;i<50;i++)
 {
  glTranslatef(0, -0.8, 0);
  bezier(ctrlPts, nCtrlPts, nBCPts);
 }
 glPopMatrix();
 glColor3f(0.7, 0.5,0.3);
 glLineWidth(5);
 glBegin(GL_LINES);
  glVertex2f(100,400);
  glVertex2f(100,40);
 glEnd();

 glutPostRedisplay();
 glutSwapBuffers();
}
void init()
{
glMatrixMode(GL_PROJECTION);
 glLoadIdentity();
 gluOrtho2D(0,500,0,500);
}



void main(int argc, char **argv)
{
 glutInit(&argc, argv);
 glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
 glutInitWindowPosition(0, 0);
 glutInitWindowSize(500,500);
 glutCreateWindow("Bezier Curve");
 init();
 glutDisplayFunc(display);
 glutMainLoop();
}
```

OUTPUT :


Bezier Curve