```
#include<iostream>
#include<stdio.h>
#include<math.h>
#include<GL/glut.h>
using namespace std;
int option = 1;
struct Point
{
GLint x;
GLint y;
};
struct Color
{
GLfloat r;
GLfloat g;
GLfloat b;
};
void init()
{
glClearColor(0.0, 0.0, 0.0, 0.0);
glColor3f(1.0, 1.0, 1.0);
glPointSize(1.0);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0,500.0,0,500.0);
}
Color getPixelColor(GLint x, GLint y)
{
Color color;
glReadPixels(x, y, 1, 1, GL_RGB, GL_FLOAT, &color);
return color;
}
void setPixelColor(GLint x, GLint y, Color color)
{
glColor3f(color.r,color.g,color.b);
glBegin(GL_POINTS);
glVertex2i(x, y);
glEnd();
glFlush();
}
void floodFill(GLint x, GLint y, Color oldColor, Color newColor)
{
Color color;
color = getPixelColor(x, y);
if(color.r == oldColor.r && color.g == oldColor.g && color.b == oldColor.b)
{
setPixelColor(x, y, newColor);
floodFill(x+1, y, oldColor, newColor);
floodFill(x, y+1, oldColor, newColor);
floodFill(x-1, y, oldColor, newColor);
```

```
floodFill(x, y-1, oldColor, newColor);
}
return;
}
void boundryFill(GLint x, GLint y, Color fColor, Color bColor)
{
Color color;
color = getPixelColor(x, y);
if((color.r !=fColor.r || color.g != fColor.g || color.b != fColor.b) &&
(color.r != bColor.r || color.g !=bColor.g || color.b !=bColor.b))
{
setPixelColor(x, y, fColor);
boundryFill(x+1, y, fColor, bColor);
boundryFill(x, y+1, fColor, bColor);
boundryFill(x-1, y, fColor, bColor);
boundryFill(x, y-1, fColor, bColor);
}
return;
}
void onMouseClick(int button, int state, int x, int y)
{
Color newColor = {0.0f, 1.0f, 0.0f};
Color oldColor = {0.0f, 0.0f, 0.0f};
Color bColor = {1.0f, 1.0, 1.0f};
Color fColor = {0.0f, 0.0f, 1.0f};
if(option == 1)
floodFill(x, 500-y, oldColor, newColor);
else
boundryFill(x, 500-y, fColor, bColor);
}
void display(void)
{
glClear(GL_COLOR_BUFFER_BIT);
glBegin(GL_LINE_LOOP); // Draw Polygon
glVertex2i(250, 250);
glVertex2i(250, 300);
glVertex2i(300, 300);
glVertex2i(300, 250);
glEnd();
glFlush();
}
int main(int argc, char** argv)
{
cout<<"1: Flood Fill"<<endl;
cout<<"2: Boundary Fill"<<endl;
cout<<"Enter Option :";
cin>>option;
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowSize(500, 500);
```

```
glutInitWindowPosition(0, 0);
glutCreateWindow("Polygon Filling");
init();
glutDisplayFunc(display),
glutMouseFunc(onMouseClick);
glutMainLoop();
return 0;
}
```