**Name – Pawar Shrikant Sanjeev**

**SE IT**

**Roll No.47**

**CG Lab**

**Assignment No.4 :Polygon Filling methods:**

**1. Flood Fill**

**2. Boundary Fill**

**-------------------------------------------------------------------------------------------------------------------------------**

**Code:**

**#include <GL/freeglut.h>**

**#include <stdio.h>**

**#include <math.h>**

**#include<iostream>**

**using namespace std;**

**struct Color {**

**float r,g,b;**

**};**

**int fillFlag;**

**void Init()**

**{**

**glClearColor(1.0,1.0,1.0,0);**

**glColor3f(0.0,0.0,0.0);**

**glPointSize(1.0);**

**glMatrixMode(GL\_PROJECTION);**

**glLoadIdentity();**

**gluOrtho2D(0 , 640 , 0 , 480);**

**}**

**void draw\_pixel(int x, int y) {**

**glBegin(GL\_POINTS);**

**glVertex2i(x, y);**

**glEnd();**

**}**

**struct Color getPixelColor(int x, int y) {**

**struct Color color;**

**glReadPixels(x, y, 1, 1, GL\_RGB, GL\_FLOAT, &color);**

**return color;**

**}**

**void setPixelColor(int x, int y,struct Color color)**

**{**

**glBegin(GL\_POINTS);**

**glColor3f(1.0, 1.0, 1.0);**

**glVertex2i(x, y);**

**glEnd();**

**//glFlush();**

**}**

**void flood(int x, int y,struct Color old\_col,struct Color new\_col)**

**{**

**struct Color color;**

**color=getPixelColor(x,y);**

**//check current pixel is old\_color or not**

**if(color.r==new\_col.r&&color.g==new\_col.g&&color.b == new\_col.b)**

**return;**

**if (color.r==old\_col.r&&color.g==old\_col.g&&color.b == old\_col.b) {**

**// put new pixel with new color**

**glBegin(GL\_POINTS);**

**glColor3f(new\_col.r,new\_col.g,new\_col.b);**

**glVertex2i(x,y);**

**glEnd();**

**glFlush();**

**// recursive call for bottom pixel fill**

**flood(x + 1, y,old\_col,new\_col);**

**// // recursive call for top pixel fill**

**flood(x - 1, y,old\_col,new\_col);**

**// // recursive call for right pixel fill**

**flood(x, y + 1,old\_col,new\_col);**

**// // recursive call for left pixel fill**

**flood(x, y - 1,old\_col,new\_col);**

**return;**

**}**

**return;**

**}**

**void boundaryFill4(int x,int y,struct Color fillColor,struct Color borderColor)**

**{**

**//float interiorColor[3];**

**struct Color color;**

**color=getPixelColor(x,y);**

**if((color.r!=borderColor.r || (color.g)!=borderColor.g || (color.b)!=borderColor.b) && (color.r!=fillColor.r || (color.g)!=fillColor.g || (color.b)!=fillColor.b))**

**{**

**glBegin(GL\_POINTS);**

**glColor3f(fillColor.r,fillColor.g,fillColor.b);**

**glVertex2i(x,y);**

**glEnd();**

**glFlush();**

**boundaryFill4(x+1,y,fillColor,borderColor);**

**boundaryFill4(x-1,y,fillColor,borderColor);**

**boundaryFill4(x,y+1,fillColor,borderColor);**

**boundaryFill4(x,y-1,fillColor,borderColor);**

**return;**

**}**

**return;**

**}**

**void createrectangle(void)**

**{**

**//to clear the buffer**

**glClear(GL\_COLOR\_BUFFER\_BIT);**

**//creating a rectangle**

**glBegin(GL\_LINE\_LOOP);**

**glVertex2f(100,200);**

**glVertex2f(100,100);**

**glVertex2f(200,100);**

**glVertex2f(200,200);**

**// glVertex2f(400,500);**

**//glVertex2f(500,400);**

**glEnd();**

**//this is the colour of the filler and old color of the object**

**struct Color old\_col={1.0,1.0,1.0};**

**struct Color new\_col={0.0,0.0,0.0};**

**struct Color fill\_col={1.0,0.0,0.0};**

**printf("fillflag: %d", fillFlag);**

**if(fillFlag==1)**

**flood(150,150,old\_col,fill\_col);**

**else if(fillFlag==2)**

**boundaryFill4(150,150,fill\_col,new\_col);**

**else if(fillFlag==3)**

**{**

**glColor3f(0.0,0.0,0.0);**

**glClear(GL\_COLOR\_BUFFER\_BIT);**

**//creating a rectangle**

**glBegin(GL\_LINE\_LOOP);**

**glVertex2f(100,200);**

**glVertex2f(100,100);**

**glVertex2f(200,100);**

**glVertex2f(200,200);**

**// glVertex2f(400,500);**

**// glVertex2f(500,400);**

**glEnd();**

**}**

**glFlush();**

**}**

**void fillMenu(int option)**

**{**

**if(option==1)**

**fillFlag=1;**

**else if(option==2)**

**fillFlag=2;**

**else if(option==3)**

**fillFlag=3;**

**}**

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

**{**

**glutInit(&argc, argv);**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);**

**glutInitWindowSize(640, 480);**

**glutInitWindowPosition(0, 0);**

**glutCreateWindow("Flood/Boundary fill");**

**glutDisplayFunc(createrectangle);**

**glutCreateMenu(fillMenu);**

**glutAddMenuEntry("Flood Fill",1);**

**glutAddMenuEntry("Boundary Fill",2);**

**glutAddMenuEntry("Empty Polygon",3);**

**glutAttachMenu(GLUT\_RIGHT\_BUTTON);**

**Init();**

**glutMainLoop();**

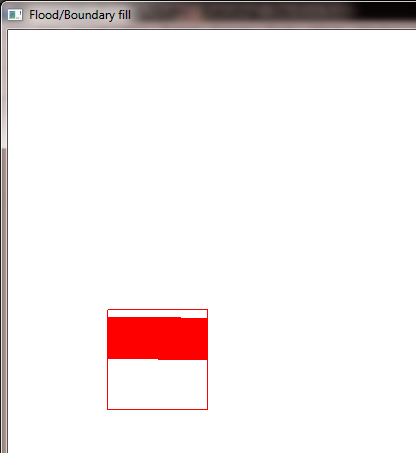
**}**

**Output:**

**Flood Fill**

****

**Boundary Fill**

****