Rajdhani
DATE / /

GROUP A Assignment 4 [CG]

Name : Aditya onkar Patil
Batch : G4 (SE4)
Roll No : 21449
Performance Date : 20/10/2021
Submission Date : 20/10/2021

TITLE: Scan Line Fill Algorithm

PROBLEM STATEMENT: Write C++ program to draw a concave polygon and fill it with desired color using scan fill algorithm. Apply the concept of inheritance.

LEARNING OBJECTIVES:
To learn the concept of scan line fill algorithm and inheritance as a concept of object oriented programming in computer graphics.

LEARNING OUTCOMES: After completion of this assignment students will be able to:
1. Implement the color filling of a concave polygon with the help of scan line fill algorithm.
2. Implement the above process with the help of the concept of inheritance.

S/W AND H/W REQUIREMENTS: 64 bit open source Linux or it's derivative, open source C++ programming tools like GCC, G++, Qt Creator, OpenGL etc.

REFERENCES: 1. Programming Principles & Practic using C++, Bjarne Stroustrup.
2. www.qt.io

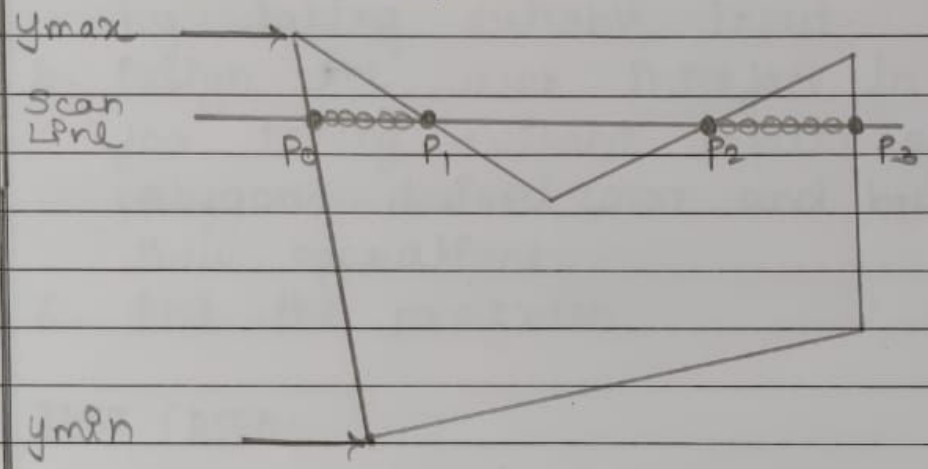
Rajdhani
DATE / /

CONCEPT RELATED THEORY:

• SCAN LINE FILL ALGORITHM:

This algorithm works by intersecting scan line with polygon edges and fills the polygon between pairs of intersections. The following steps depict how the algorithm works.

1. Find out the y_{min} and y_{max} from the given polygon.



2. Scanline intersects with each edge of the polygon from ymin into ymax. Name each intersectionpoint of the polygon.
3. Sort the intersection point in increasing order of x coordinate i.e. p0, p1, p1, p2, p2, p3,.
4. Fill all these pair of coordinates that are inside polygons and ignore the alternate pairs.

• INHERITANCE: It can be described as a process of creating new classes from existing classes. New classes inherit some of the properties and behaviour of existing classes and hence are called derived classes. Syntax:

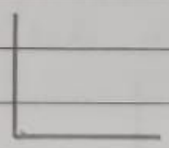
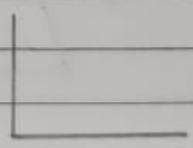
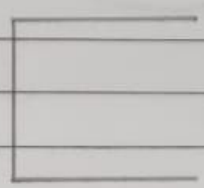
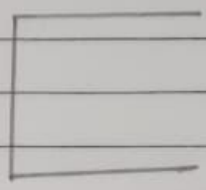
Class derived class: access specifier Base Class

Rajdhani
DATE / /

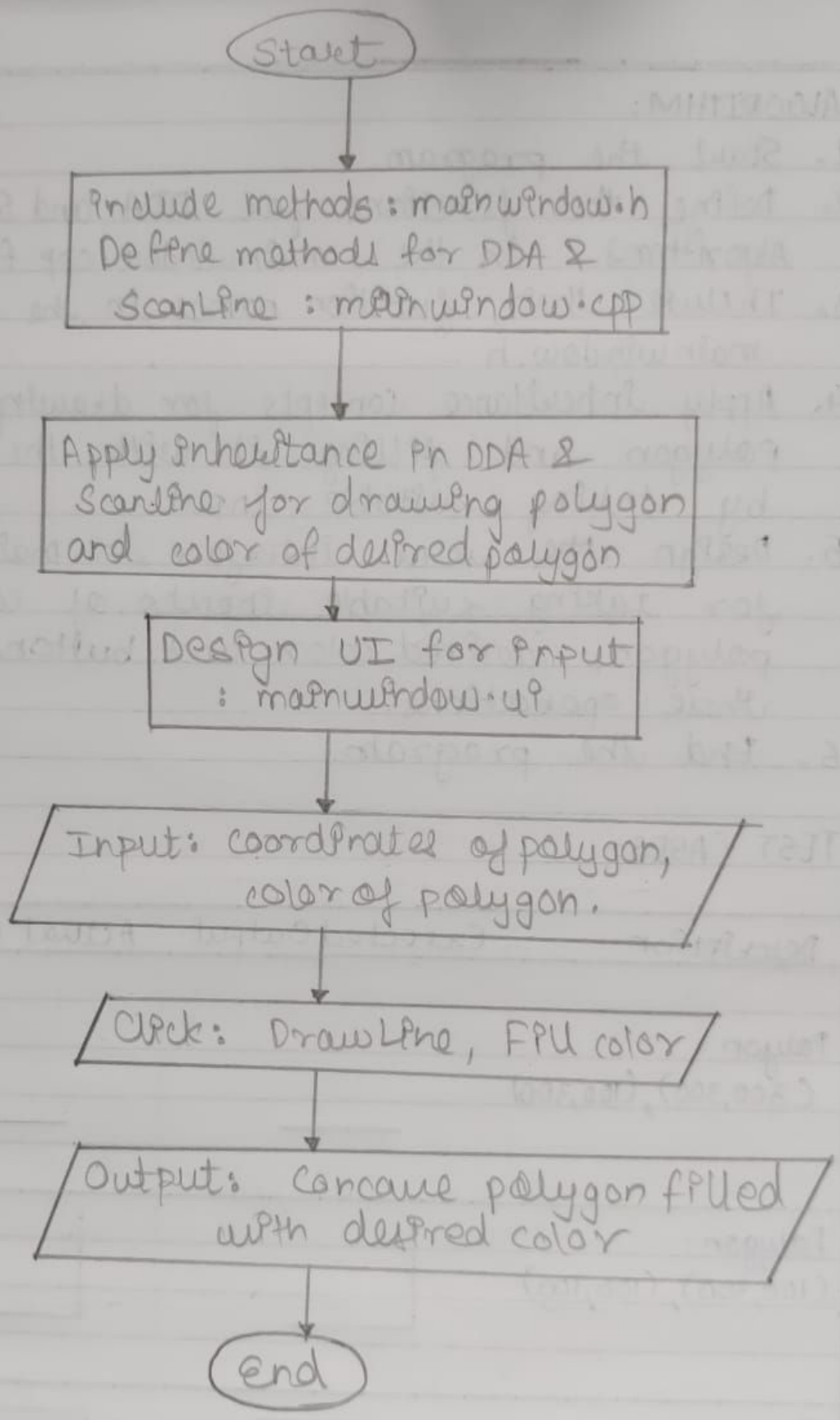
ALGORITHM:

1. Start the program.
2. Define the functions for DDA and Scan Line Algorithms in the mainwindow.cpp file.
3. Include their function names in the header file mainwindow.h.
4. Apply inheritance concepts for drawing a concave polygon and filling it with the desired color by taking suitable input.
5. Design the user interface in mainwindow.ui for taking suitable inputs of coordinates of polygon, desired color and buttons for executing these operations.
6. End the program.

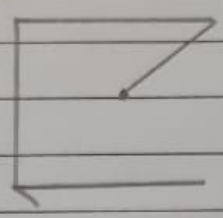
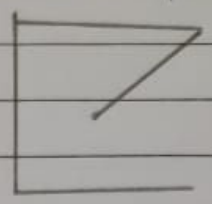
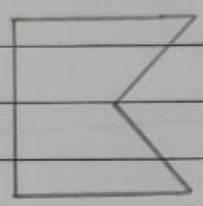
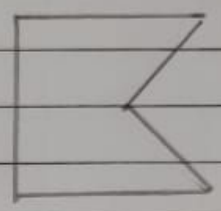
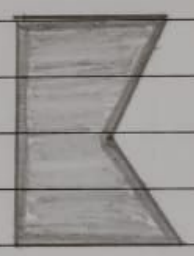
TEST CASES:

Sr. No.	Description	Expected Output	Actual Output	Status
1.	Polygon: (300,300), (100,300)			Pass
2.	Polygon: (100,300), (100,100)			Pass
3.	Polygon: (100,100), (300,100)			Pass

FLOW CHART:


Start
Include methods : mainwindow.h
Define methods for DDA &
ScanLine : mainwindow.cpp
Apply inheritance in DDA &
Scanline for drawing polygon
and color of desired polygon
Design UI for input
: mainwindow.ui
Input: coordinates of polygon,
color of polygon.
Click: DrawLine, Fill color
Output: Concave polygon filled
with desired color
End

Rajdhani
DATE / /

Sr. No.	Description	Expected Output	Actual Output	Status
4.	Polygon: (300,100), (200,200)			Pass
5.	Polygon: (200,200), (300,300)			Pass
6.	Fill Polygon: Select color & click Fill			Pass

CONCLUSION:

The concepts of Scan Line Fill Algorithm for colouring concave polygons were successfully understood and implemented using concepts of inheritance.

mainwindow.h @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- mainwindow.cpp
- mainwindow.h

mainwindow.h | <Select Symbol> | Windows (CRLF) | Line: 40, Col: 1

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QMessageBox>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();
    void DDALine(float,float,float,float,int);
    void ScanLineFill();
   // void mousePressEvent(QMouseEvent *);

private slots:


    void on_pushButton_clicked();

    void on_pushButton_3_clicked();

    void on_pushButton_4_clicked();

    void on_pushButton_2_clicked();

    //void on_label_linkActivated(const QString &link);

    void on_pushButton_5_clicked();

private:
    Ui::MainWindow *ui;
};
```

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results

main.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

main.cpp <No Symbols> Windows (CRLF) Line: 1, Col: 1

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();
    return a.exec();
}
```

Welcome | Edit | Design | Debug | Projects | Help

unti..._CG2 Debug

Type to locate (Ctrl... 1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results

mainwindow.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

mainwindow.cpp | MainWindow::ScanLineFill() -> void | Windows (CRLF) | Line: 150, Col: 15

```cpp
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QColorDialog>
#include <math.h>
#include <QTime>
#define height 500
#define width 500
using namespace std;

int vert,i,y,xi[20],yi[20],j,temp,k;
int a[20][2];
float slope[20],dx,dy;



QImage img(width, height, QImage::Format_RGB888);
QRgb rgb(qRgb(255,255,255));

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    for (int x = 0; x < width; ++x)
        {
            for (int y = 0; y < height; ++y)
            {
                img.setPixel(x, y, qRgb(0, 0, 0));      //setting all the pixels on the screen to black (0,0,0)
            }
        }
     ui->label->setPixmap(QPixmap::fromImage(img));
     vert=1;

}

MainWindow::~MainWindow()
{
    delete ui;
}
```

1 Issues | 2 Search Results | 3 Application Output | 4 Compile Output | 5 QML Debugger Console | 6 General Messages | 8 Test Results

mainwindow.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

mainwindow.cpp | MainWindow::ScanLineFill() -> void | Windows (CRLF) | Line: 150, Col: 15

```cpp

void MainWindow::on_pushButton_4_clicked(){
  int p=ui->textEdit->toPlainText().toFloat();
  int q=ui->textEdit_2->toPlainText().toFloat();

  a[0][0]=300;                          // first vertex is set to (300,300)
  a[0][1]=300;

  a[vert][0]=p;
  a[vert][1]=q;
  DDALine(a[vert][0],a[vert][1],a[vert-1][0],a[vert-1][1],0);
  vert++;


;
}
void delay( int millisecondsToWait )
{
    QTime dieTime = QTime::currentTime().addMSecs( millisecondsToWait );
    while( QTime::currentTime() < dieTime )
    {
        QCoreApplication::processEvents( QEventLoop::AllEvents, 100 );
    }
}

void MainWindow::DDALine(float x1, float y1, float x2, float y2, int delay_time=1)
{

    float dx = (x2 - x1);
    float dy = (y2 - y1);
    float step;

    if (abs(dx)>=abs(dy))
    {
        step = abs(dx);
    }
    else
    {
        step = abs(dy);
    }
```

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results

mainwindow.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

mainwindow.cpp | MainWindow::ScanLineFill() -> void | Windows (CRLF) | Line: 150, Col: 15

```cpp
        delay(delay_time);
    }


}

void MainWindow::ScanLineFill()
{   int ymax=0,ymin=10000000;
    a[0][0]=300;
    a[0][1]=300;
    a[vert][0]=a[0][0];
    a[vert][1]=a[0][1];
    for(i=0;i<vert-1;i++)
                {
                        if(a[i][1]>ymax){
                            ymax=a[vert][i];
                        }
                        if(a[i][1]<ymin){
                            ymin=a[vert][i];
                        }

                }
    for(i=0;i<vert;i++){
                dx=a[i+1][0]-a[i][0];
                dy=a[i+1][1]-a[i][1];
                if(dx==0.0){slope[i]=1.0;}
                if(dy==0.0){slope[i]=0.0;}
                if(dx!=0.0 and dy!=0.0){slope[i]= float(dx/dy);}
                }
    for(int y=0;y<500;y++){
        int k=0;
        for(i=0;i<vert;i++){
            if((y>=a[i][1] and y<a[i+1][1])or (y>=a[i+1][1]and y<a[i][1])){
                xi[k]= a[i][0]+(slope[i]*(y-a[i][1]));
                yi[k]= a[i][1];
                k++;
            }
        }
        for(i=0;i<k-1;i++){
            for(j=0;j<k-1-i;j++){
                if(xi[j]>xi[j+1]){
```

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results

mainwindow.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

mainwindow.cpp | MainWindow::ScanLineFill() -> void | Windows (CRLF) | Line: 150, Col: 15

```cpp
                if(xi[j]>xi[j+1]){
                    temp=xi[j];
                    xi[j]=xi[j+1];
                    xi[j+1]=temp;
                }
            }
        }
        for(j=1;j<k;j++){
            DDALine(xi[j],y,a[j][0],y);

            //for(int z=1;z<k;z++) {
              //  DDALine(xi[j],yi[z],xi[j],yi[z]);

            //}

        }
    }
}


void MainWindow::on_pushButton_clicked()                                //COLOR
{
    QRgb color(QColorDialog::getColor().rgb());
    rgb = color;
}


void MainWindow::on_pushButton_3_clicked()
{
    QMessageBox message;
    message.information(0,"Screen has been cleared.","Please draw new polygon"); //RESET
    for (int x = 0; x < width; ++x)
        {
            for (int y = 0; y < height; ++y)
            {
                img.setPixel(x, y, qRgb(0, 0, 0));
            }
        }


        ui->label->setPixmap(QPixmap::fromImage(img));
```

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results

mainwindow.cpp @ untitled_CG2 - Qt Creator

File Edit View Build Debug Analyze Tools Window Help

Projects

- untitled_CG2
  - untitled_CG2.pro
  - Headers
    - mainwindow.h
  - Sources
    - main.cpp
    - mainwindow.cpp
  - Forms

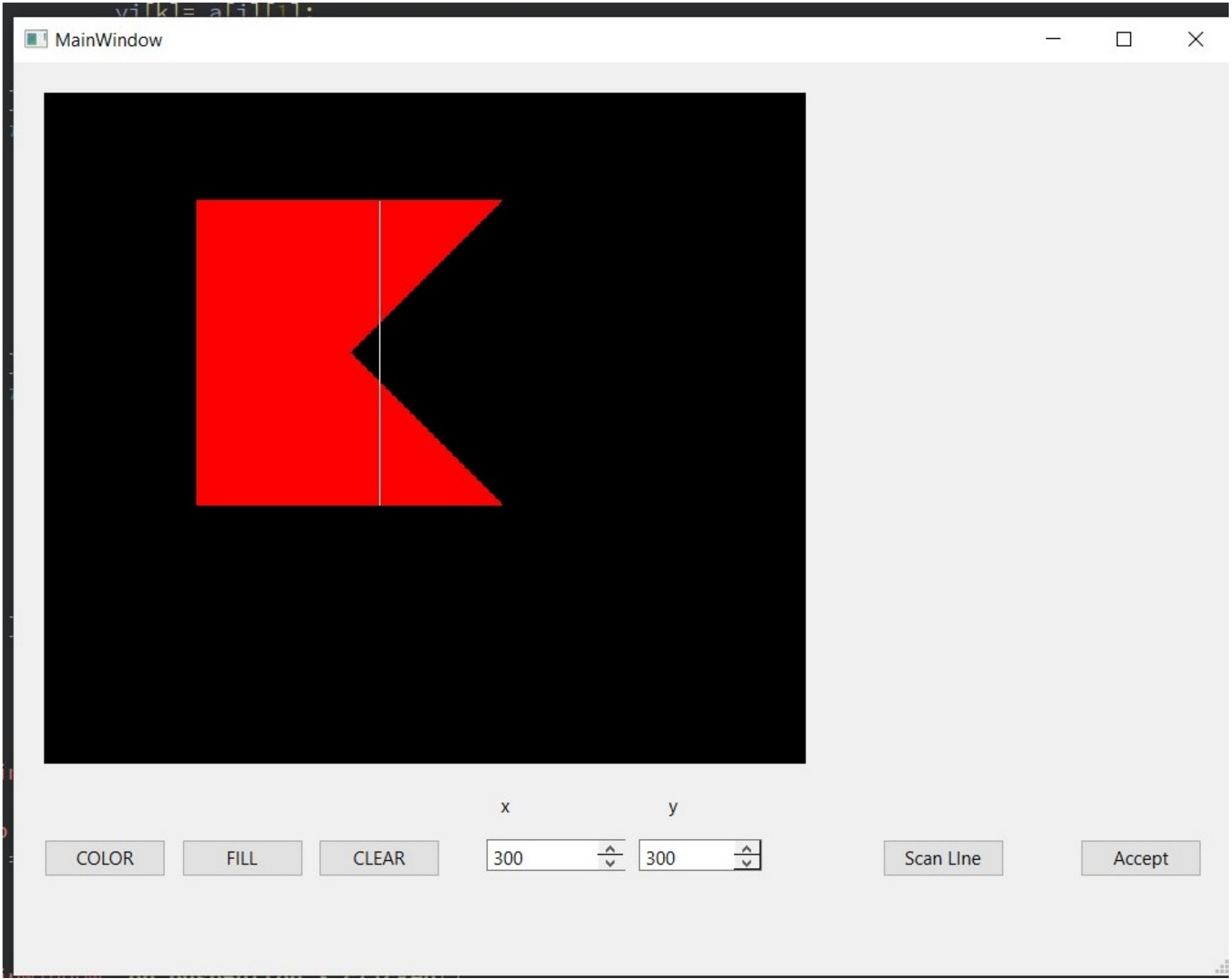
Open Documents

- main.cpp
- mainwindow.cpp
- mainwindow.h

mainwindow.cpp | MainWindow::ScanLineFill() -> void | Windows (CRLF) | Line: 150, Col: 15

```cpp
        {
            for (int y = 0; y < height; ++y)
            {
                img.setPixel(x, y, qRgb(0, 0, 0));
            }
        }


        ui->label->setPixmap(QPixmap::fromImage(img));
        for (int q =0; q<=vert;q++)
        {
            for (int b=0;b<2;b++)
            {
                a[q][b]=0;
            }
        }
        vert=1;

}

void MainWindow::on_pushButton_2_clicked()                    //FILL
{
    QMessageBox message;
    if(a[0][0]==a[vert-1][0] && a[0][1]==a[vert-1][1])
    {
        ScanLineFill();
        for (int q =0; q<=vert;q++)
        {
            for (int b=0;b<2;b++)
            {
                a[q][b]=0;
            }
        }
        vert=1;

    }
    else
    {
        message.information(0,"Warning!","The polygon has not been closed properly! Please check the coordinates!");
     }
```

1 Issues 2 Search Results 3 Application Output 4 Compile Output 5 QML Debugger Console 6 General Messages 8 Test Results


MainWindow
x
y
COLOR
FILL
CLEAR
300
300
Scan LIne
Accept