- **Source Code**

- ***MainWindow.h***

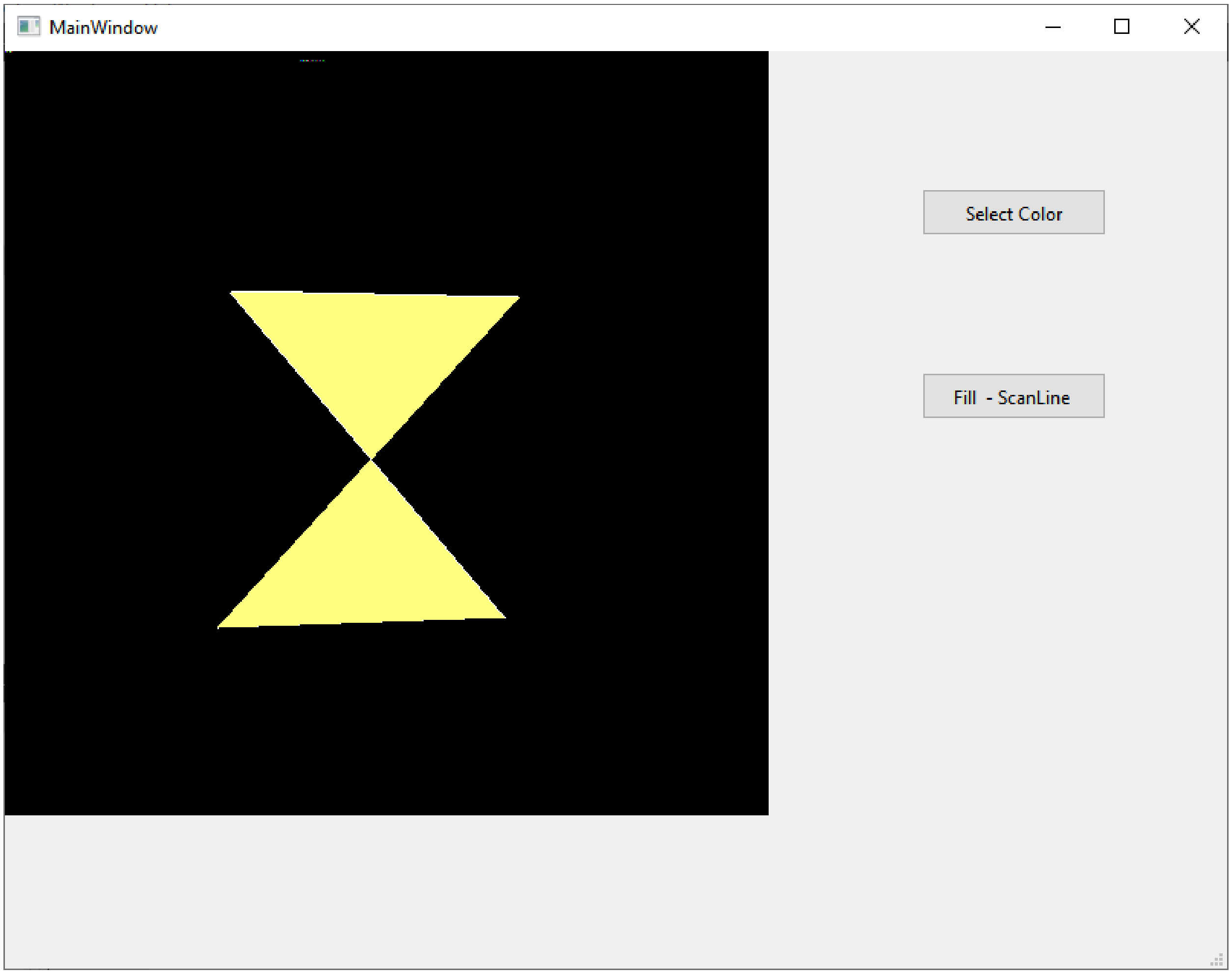
```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>

QT_BEGIN_NAMESPACE
namespace Ui {
class MainWindow;
}
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
    // QT creator function definitions
    void on_pushButton_clicked();
    void on_pushButton_2_clicked();

    // Our function definitions
    void DDA(float, float, float, float);
    void mousePressEvent(QMouseEvent *event);

private:
    Ui::MainWindow *ui;
    bool start;
    int count, ver, temp, i, j , k, m, n;
    int a[100], b[100], xi[20];
    float dy, dx, slope[20];
};
#endif // MAINWINDOW_H
```

- *MainWindow.cpp*

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <QColorDialog>
#include <QTime>
#include <QMouseEvent>
#include <iostream>
using namespace std;
QColor color = qRgb(255,255,255);
QImage img(500,500,QImage::Format_RGB888);
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    start = true;
    count = 0;
    ui->setupUi(this);
}

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

void delay(int delayTime)
{
    QTime waitTime = QTime::currentTime().addMSecs(delayTime);
    while(QTime::currentTime() < waitTime)
    {
        QCoreApplication::processEvents(QEventLoop::AllEvents, 400);
    }
}

void MainWindow::DDA(float x1, float y1, float x2, float y2)
{
    float dx = x2-x1;
    float dy = y2-y1;
    float length = (abs(dx) > abs(dy)) ? abs(dx) : abs(dy);
    float xinc = dx/length;
    float yinc = dy/length;
    int i = 0;
    while(i<length)
    {
        img.setPixel(x1, y1, color.rgb());
        x1 += xinc;
        y1 += yinc;
        i++;
    }
    ui->label->setPixmap(QPixmap::fromImage(img));
}

void MainWindow::mousePressEvent(QMouseEvent *event)
{
    if(start)
    {
        a[count] = event->pos().x();
        b[count] = event->pos().y();

        if(event->button() == Qt::RightButton)
        {
            DDA(a[count-1], b[count-1], a[0], b[0]);
            start = false;
            cout << "Polygon drawn\n";
        }
        else
        {
            if(count > 0)
            {
                DDA(a[count], b[count], a[count-1], b[count-1]);
            }
            count++;
        }
    }
}

void MainWindow::on_pushButton_clicked()
{
    // Select color from user
    color = QColorDialog::getColor();
}
```

```
void MainWindow::on_pushButton_2_clicked()
{
    // | ymax or current y | (x) of ymin | 1/slope |
    a[count] = a[0];
    b[count] = b[0];

    for(i = 0; i<count; i++)
    {
        dy = b[i+1] - b[i];
        dx = a[i+1] - a[i];

        if(dy == 0.0f)
            slope[i] = 1;
        else if(dx == 0.0f)
            slope[i] = 0.0;
        else
            slope[i] = dx / dy; // 1/slope = 1/(dy/dx)
    }
    for(m=0; m<500; m++)
    {
        k=0;
        for(i=0; i<count; i++)
        {
            if((b[i] <= m && b[i+1] > m) || (b[i] > m && b[i+1] <= m))
            {
                // y = mx + c
                xi[k] = int(a[i] + slope[i]*(m-b[i]));
                k++;
            }
        }

        for(j=0; j<k-1; j++)
        {
            for(n = 0; n< k-j-1; n++)
            {
                if(xi[n+1] > xi[n])
                {
                    // Swap
                    temp = xi[i];
                    xi[i] = xi[i+1];
                    xi[i+1] = temp;
                }
            }
        }

        for(i=0;i<k; i+=2)
        {
            delay(20);
            DDA(xi[i], m, xi[i+1], m);
        }
    }
}
```

- **Output**


MainWindow
Select Color
Fill - ScanLine