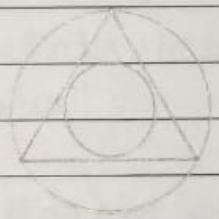
Rajdhani
DATE / /

Group A Assignment 6 [CG]

Name : Aditya Onkar Patil
Batch : G4 (SE4)
Roll No : 21449
Performance Date : 6/10/21
Submission Date : 9/10/21

TITLE: DDA line drawing & Bresenham's Circle Algorithm

PROBLEMSTATEMENT: Write a C++ program to draw the following pattern. Use DDA line and Bresenham's circle drawing algorithm. Apply the concept of encapsulation.



LEARNING OBJECTIVES:

To learn the concept of DDA line drawing algorithm and Bresenham's circle drawing algorithm along with the usage of data encapsulation.

LEARNING OUTCOMES: After completion of this assignment students will be able to:

1. Implement the drawing of various two-dimension figures using DDA line drawing and Bresenham's circle drawing algorithms.
2. Implement the above functions in an efficient manner by use of concept of data encapsulation.

S/W and H/W requirements: 64 bit open source Linux or its derivative, open source C++ programming tools like GCC, G++, Qt Creator, OpenGL etc.

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

CONCEPT RELATED THEORY:

- DDA LINE DRAWING ALGORITHM: DDA is the simplest line drawing Algorithm. Given the starting and ending coordinates of a line, DDA algorithm attempts to generate the points between the starting and ending coordinates.

Algorithm:

1. Given starting and ending coordinates: (x_1, y_1), (x_2, y_2)
2. Calculate dx and dy: $dx = x_2 - x_1$
   $dy = y_2 - y_1$
3. Compare the absolute value of dx and dy and assign the bigger one to a variable step.
4. Define $x_{increament} = dx / step$
   $y_{increament} = dy / step$
5. Set the current pixel $x = x_1$.
   $y = y_1$.
6. Increament the pixel $x = x_1 + x_{increament}$
   $y = y_1 + y_{increament}$
   until the point (x_2, y_2) is attained.

Rajdhani
DATE / /

Bresenham's Circle Drawing Algorithm:
This algorithm's key feature is the symmetry that a circle exhibits. Here, the circle will be divided into 8 points (octants) for setting pixels which in turn, finally draw a circle.

Algorithm:

1. Declare p, q, x, y, r, d variables
   p, q are the coordinates of the centre of the circle, r is the radius of the circle
2. Calculate $d = 3 - 2r$
3. Initialize $x = 0$, $y = r$
4. Check if whole circle is scan converted
   If $x >= y$, stop the program.
5. Plot eight points by using concepts of eight way symmetry. The centre is at (p, q). Current pixel is (x, y)
   putpixel (x+p, y+q)
   putpixel (y+p, x+q)
   putpixel (-y+p, x+q)
   putpixel (-x+p, y+q)
   putpixel (-x+p, -y+q)
   putpixel (-y+p, -x+q)
   putpixel (y+p, -x+q)
   putpixel (x+p, -y-q)
6. Find location of next pixel to be scanned,
   If $d < 0$; $d = d + 4x + 6$ & increament $x = x + 1$
   If $d \geq 0$; $d = d + 4(x - y) + 10$ & increament $x = x + 1$
   & decreament $y = y - 1$
7. Go to step 4.
8. Stop Algorithm.

Rajdhani
DATE / /

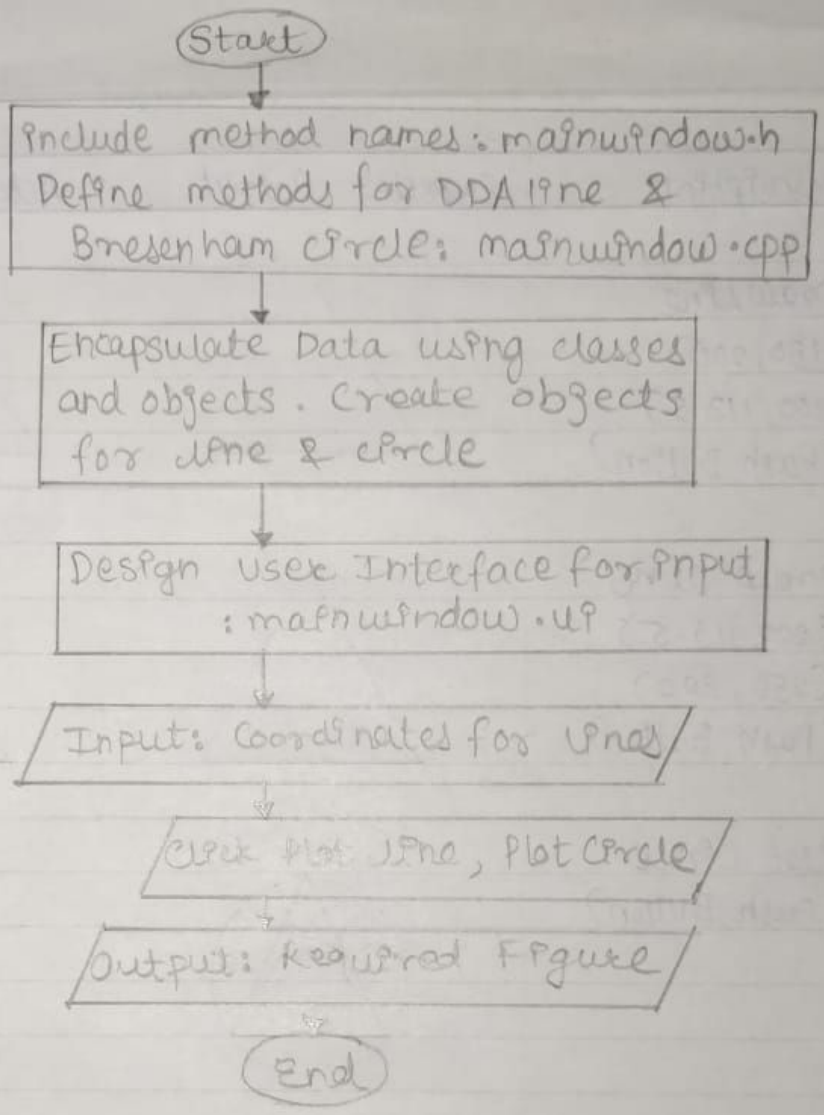
ALGORITHM:

1. Start the program.
2. Define the functions for DDA and Bresenham's circle drawing algorithm in the mainwindow.cpp file.
3. Include their function names in the header file mainwindow.h
4. Encapsulate the data using classes and create objects for drawing a line via 4 parameters (x_1, y_1) and (x_2, y_2) as starting and ending point of a line.
5. Take input in such a way that it forms an equilateral triangle.
6. Create objects for drawing a circle via 2 parameters (x_1, y_1) as the centre of the circle.
7. Calculate the radii of two circles using geometry and pass them as parameters in the Bresenham's circle drawing function.
8. Design the user interface in mainwindow.ui to take input from the user and display output to the user.
9. End the program.

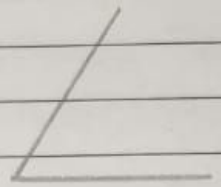
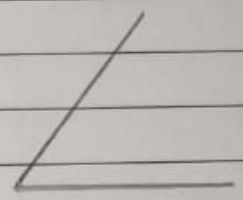
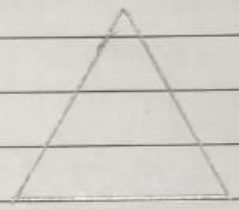
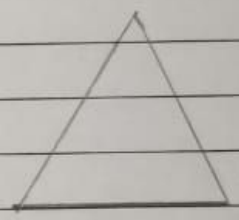
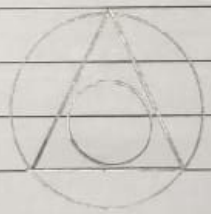
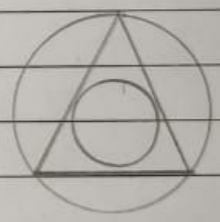
TEST CASES:

No.	Description	Expected Output	Actual output	Status
1.	Draw line (150, 200) (250, 200) (Push Button)	———	———	Pass

FLOWCHART:


Start
Include method names: mainwindow.h
Define methods for DDA line &
Bresenham circle: mainwindow.cpp
Encapsulate Data using classes
and objects. Create objects
for line & circle
Design User Interface for input
: mainwindow.ui
Input: Coordinates for lines
Click Plot line, Plot Circle
Output: Required Figure
End

Rajdhani
DATE / /

Sr. No.	Description	Expected Output	Actual Output	Status
2.	Draw Line (150, 200) (200, 113.5) (Push Button)			Pass
3.	Draw line (200, 113.5) (250, 200) (Push Button)			Pass
4.	Plot Circle (Push Button)			Pass

CONCLUSION :-

The knowledge of DDA line drawing algorithm and Bresenham's circle drawing algorithm was successfully acquired and implemented using the concept of encapsulation.

Tools Window Help

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

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QPainter>

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    void DDA(float x1, float y1, float x2, float y2);
    void BresCirc(int xc, int yc, int r);
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
    void on_pushButton_clicked();

    void on_pushButton_2_clicked();

private:
    Ui::MainWindow *ui;
};
#endif // MAINWINDOW_H

```

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

Tools Window Help

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

```cpp
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;                           // Object of Main Window
    w.setWindowTitle("21449_OOPCGL_CG1");
    w.show();
    return a.exec();
}
```

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

Tools Window Help

mainwindow.cpp <Select Symbol> Windows (CRLF) Line: 1, Col: 1

```cpp
#include "mainwindow.h"
#include "ui_mainwindow.h"

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

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

QImage img(400,400,QImage::Format_RGB888);                  // Image for output  (Black Background)

float  xcen = 0, ycen = 0;                                  // Variables for centroid

void MainWindow::DDA(float x1, float y1, float x2, float y2){      // Function for line drawing (DDA)
    float dx, dy, length, Xinc, Yinc;
    xcen = xcen + x1 + x2; ycen = ycen + y1 + y2;
    dx = (x2 - x1); dy = (y2 - y1);
    if (abs(dx)>abs(dy)){
        length = abs(dx);                                   // (x1+x2+x3)/3
    }
    else{
        length = abs(dy);
    }
    Xinc = dx/length; Yinc = dy/length;
    for(int i = 0;i <=length; i++){
       img.setPixel(x1,y1,qRgb(225,225,225));
       x1 = x1 + Xinc; y1 = y1 + Yinc;
    }
}

void MainWindow::on_pushButton_clicked()              // Functions to carry out when pressed "Plot Triangle"
{
    float x1, y1, sidelength,x2,y2;
    x1 = ui->textEdit_x1->toPlainText().toFloat();
    y1 = ui->textEdit_y1->toPlainText().toFloat();
    x2 = ui->textEdit_x2->toPlainText().toFloat();
```

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

Tools Window Help

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

```cpp

void MainWindow::on_pushButton_clicked()              // Functions to carry out when pressed "Plot Triangle"
{
    float x1, y1, sidelength,x2,y2;
    x1 = ui->textEdit_x1->toPlainText().toFloat();
    y1 = ui->textEdit_y1->toPlainText().toFloat();
    x2 = ui->textEdit_x2->toPlainText().toFloat();
    y2 = ui->textEdit_y2->toPlainText().toFloat();
    sidelength = ui->textEdit_sl->toPlainText().toFloat();    ⚠ Value stored to 'sidelength' is never read [clang-analyzer-deadcode.DeadS…
    DDA(x1,y1,x2,y2);
    ui->label_5->setPixmap(QPixmap::fromImage(img));
}
void MainWindow::BresCirc(int xc, int yc, int r){       // Function for circle (Bresenham)
    int x, y, d;
    x = 0; y = r;
    img.setPixel(x+xc,y+yc,qRgb(225,225,225));
    d = 3 - (2*r);
    while(x<=y){
        if(d<0){
            d = d + (4*x) + 6;
            x++;
        }
        else{
            d = d + 4*(x-y) + 10;
            x++; y--;
        }
        img.setPixel(xc+x, yc+y, qRgb(225,225,225));
        img.setPixel(xc-x, yc+y, qRgb(225,225,225));
        img.setPixel(xc+x, yc-y, qRgb(225,225,225));
        img.setPixel(xc-x, yc-y, qRgb(225,225,225));
        img.setPixel(xc+y, yc+x, qRgb(225,225,225));
        img.setPixel(xc-y, yc+x, qRgb(225,225,225));
        img.setPixel(xc+y, yc-x, qRgb(225,225,225));
        img.setPixel(xc-y, yc-x, qRgb(225,225,225));
    }
}

void MainWindow::on_pushButton_2_clicked()
{                                                       // Functions to carry out when pressed "Plot circle"
    float xcemtre = xcen/6;
    float ycemtre = ycen/6;
    float sidelength = ui->textEdit_sl->toPlainText().toFloat();
```

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

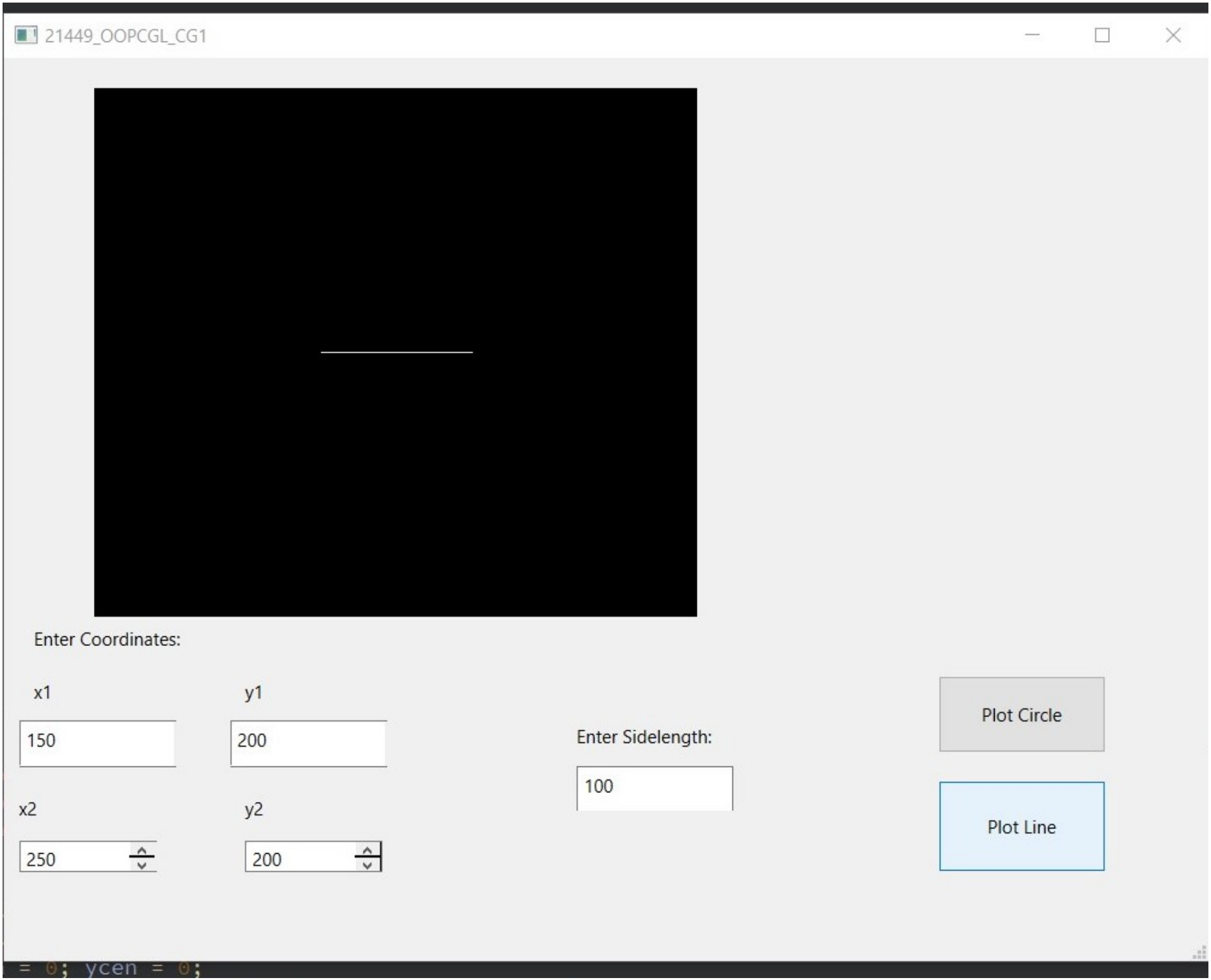
Tools Window Help

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

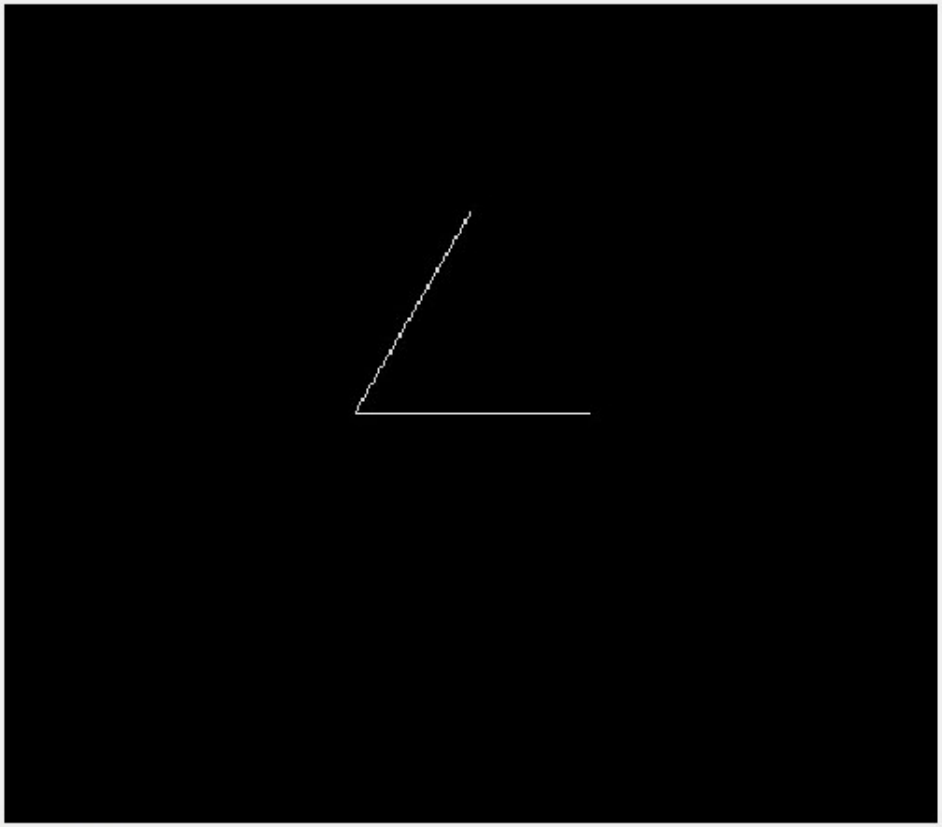
```
    ui->label_5->setPixmap(QPixmap::fromImage(img));
}
void MainWindow::BresCirc(int xc, int yc, int r){       // Function for circle (Bresenham)
    int x, y, d;
    x = 0; y = r;
    img.setPixel(x+xc,y+yc,qRgb(225,225,225));
    d = 3 - (2*r);
    while(x<=y){
        if(d<0){
            d = d + (4*x) + 6;
            x++;
        }
        else{
            d = d + 4*(x-y) + 10;
            x++; y--;
        }
        img.setPixel(xc+x, yc+y, qRgb(225,225,225));
        img.setPixel(xc-x, yc+y, qRgb(225,225,225));
        img.setPixel(xc+x, yc-y, qRgb(225,225,225));
        img.setPixel(xc-x, yc-y, qRgb(225,225,225));
        img.setPixel(xc+y, yc+x, qRgb(225,225,225));
        img.setPixel(xc-y, yc+x, qRgb(225,225,225));
        img.setPixel(xc+y, yc-x, qRgb(225,225,225));
        img.setPixel(xc-y, yc-x, qRgb(225,225,225));
    }
}

void MainWindow::on_pushButton_2_clicked()
{                                                       // Functions to carry out when pressed "Plot circle"
    float xcemtre = xcen/6;
    float ycemtre = ycen/6;
    float sidelength = ui->textEdit_sl->toPlainText().toFloat();
    int smallr = sidelength/(3.46);
    int bigr = sidelength/(1.73);
    BresCirc(xcemtre, ycemtre, smallr);
    BresCirc(xcemtre, ycemtre, bigr);
    xcen = 0; ycen = 0;
    ui->label_5->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


21449_OOPCGL_CG1
Enter Coordinates:
x1
150
y1
200
x2
250
y2
200
Enter Sidelength:
100
Plot Circle
Plot Line
= 0; ycen = 0;

21449_OOPCGL_CG1



Enter Coordinates:

x1

150

y1

200

x2

200

y2

113.5

Enter Sidelength:

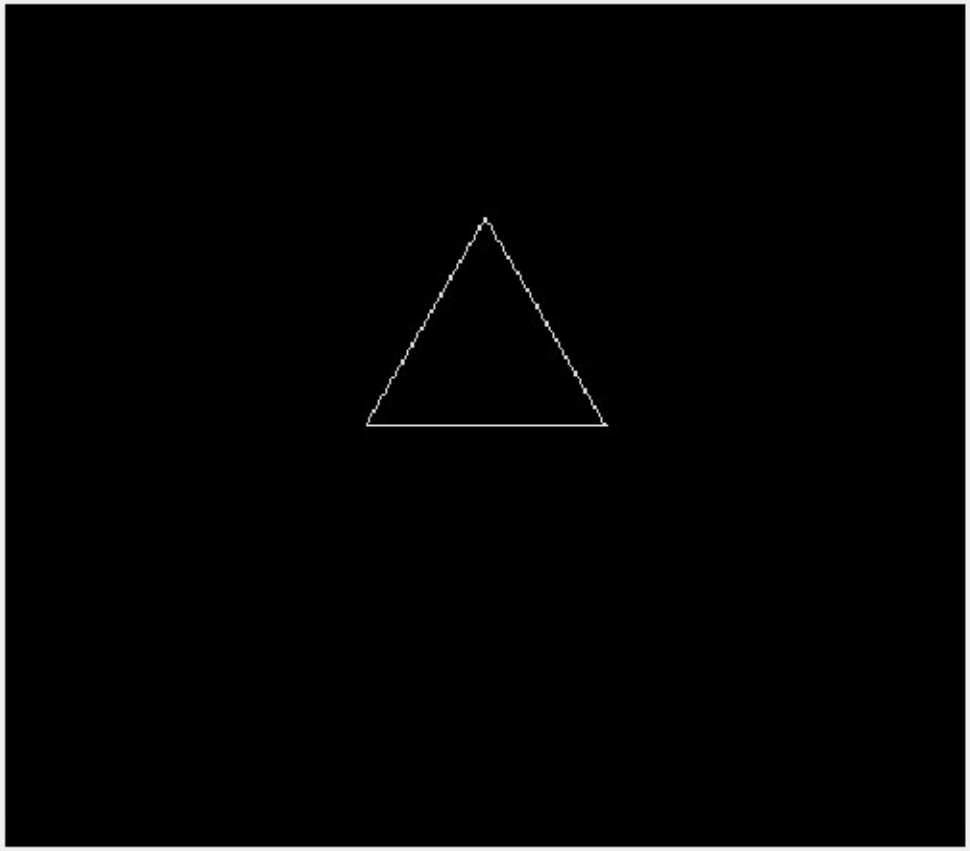
100

Plot Circle

Plot Line

```
= 0; ycen = 0;
```

21449_OOPCGL_CG1



Enter Coordinates:

x1

250

y1

200

x2

200

y2

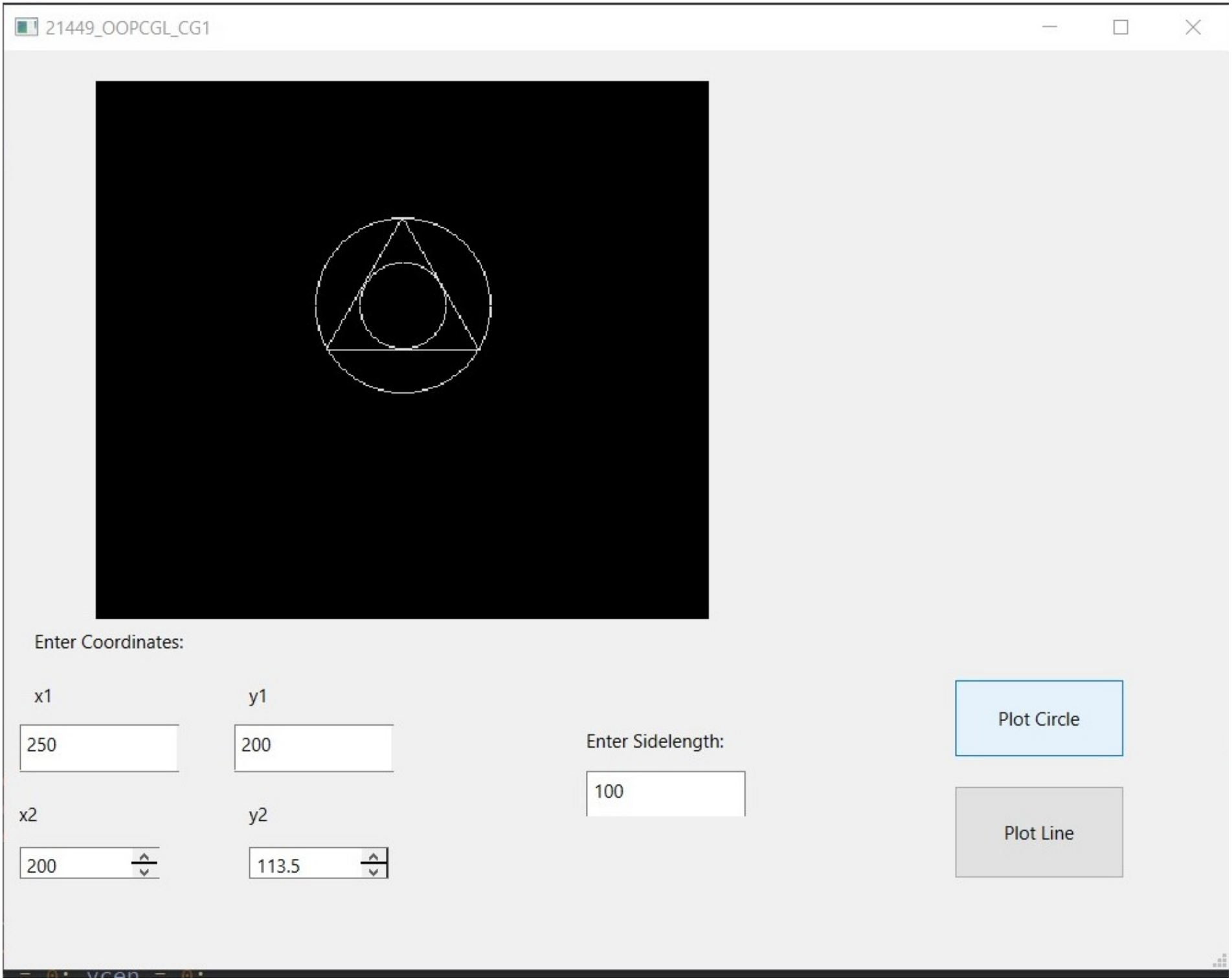
113.5

Enter Sidelength:

100

Plot Circle

Plot Line


21449_OOPCGL_CG1
Enter Coordinates:
x1
250
y1
200
x2
200
y2
113.5
Enter Sidelength:
100
Plot Circle
Plot Line