Assignment 10

Class- SE IV ROll NO-21430
Batch - F4
D.O.S - 6/11/2020

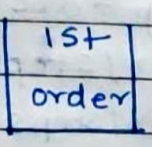
Problem statement -:
a) Write c++ program to generate snowflake using concept of fractals or
b) Hilbert curve or
c) Koch curve.

Learning Objectives -:
To study curves and fractals.

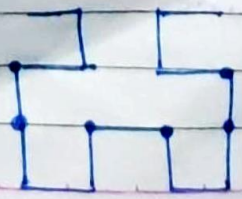
Theory -:
The Hilbert Curve -:
The hilbert curve is a space filling curve that visits every point in a square grid with a size of 2×2, 4×4, 8×8, 16×16 or any other power of 2. It was first described by David Hilbert in 1892. Applications of the Hilbert curve are in image processing especially image compression. It has advantages in these operations where the coherence between neighbouring pixels is important. The Hilbert curve is also a special version of a quadtree, any

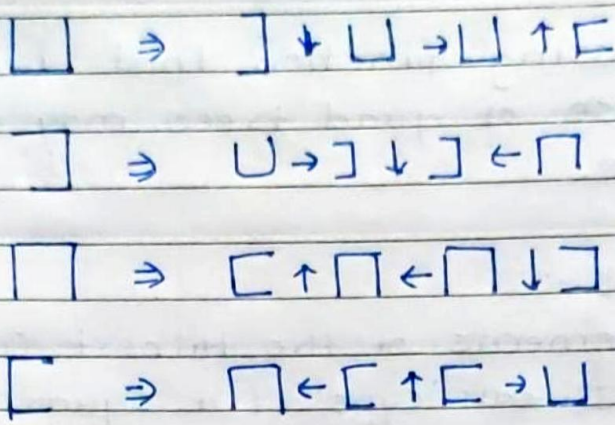
image processing function that benefits from the use of quad trees may also use Hilbert Curve.

Cyps and Joint :-
The basic elements of the Hilbert Curve are what I call 'cyps" (a square with one open side) and "joins" (a vector that joints two cyps). The "open" side of a cyp can be top bottom, left or right. In addition every cyp has two end-points and each of these can be the entry point or the exit point so there are eight possible varieties of cyps. In practice a Hilbert curve uses only four types of cyps. In a similar vein, a join has a direction : up, down left or down.



A first order Hilbert curve is just a single cup (see the fig. on the left). It fills a 2x2 space. The second order Hilbert curve replaces that cyp by four (smaller) Cyps which are linked together by three joins. the link bet n a cyp and a join has been marked with





* Psuedocode

* Start

def dda (int x, int y , int x_2, int y_2)

1. Read x_1, y_1, x_2, y_2.
2. set $dx = x_2 - x_1$
   set $dy = y_2 - y_1$
3. if ($dx > dy$)
   3.1. set step = abs (dx)
   else
   3.2. set step = abs (dy)
4. Set x_{inc} = dx/steps
   Set y_{inc} = dy/steps
5. Initialize $x = x_1$
   Initialize $y = y_1$
6. Set i = 0
7. for (i = 0 , i <steps ; i++)
   Setpixel (x, y, colour)
   $x = x + x_{inc}$
   $y = y + y_{inc}$.

END.

def Move (int j, int h, int &x, int &y)

1. set $x_1 = x$

   set $y_1 = y$

2. switch (j)

   if (j==1)

   set y = y-h

   if (j==2)

   set x = x+h

   if (y==3)

   set y = y+h

   if (y==4)

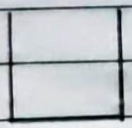
   set x = x-h

3. call dda_function (x_1, y_1, x, y)

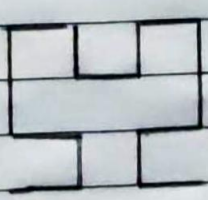
END

* Dry Run

i) Order - 1

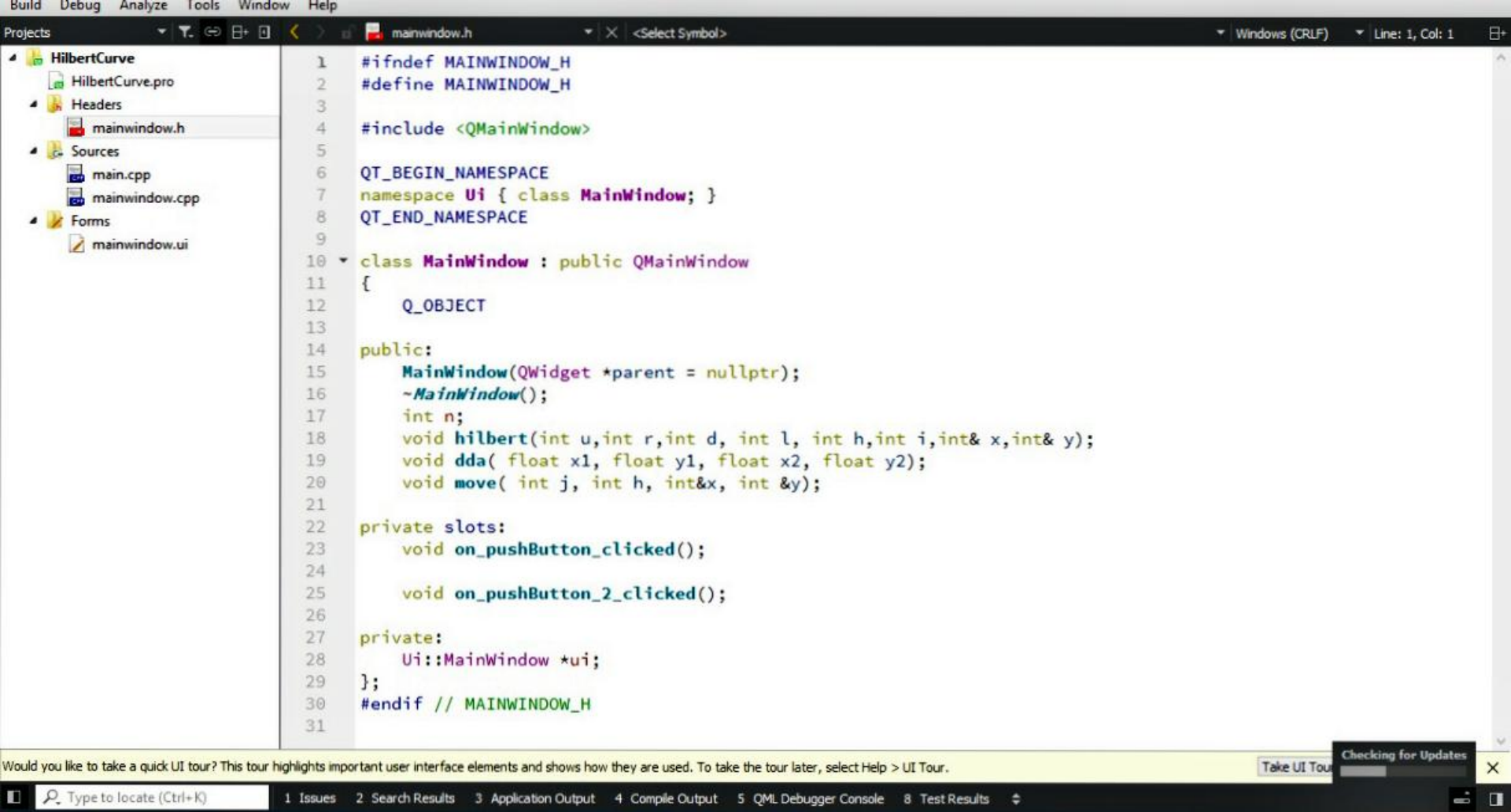


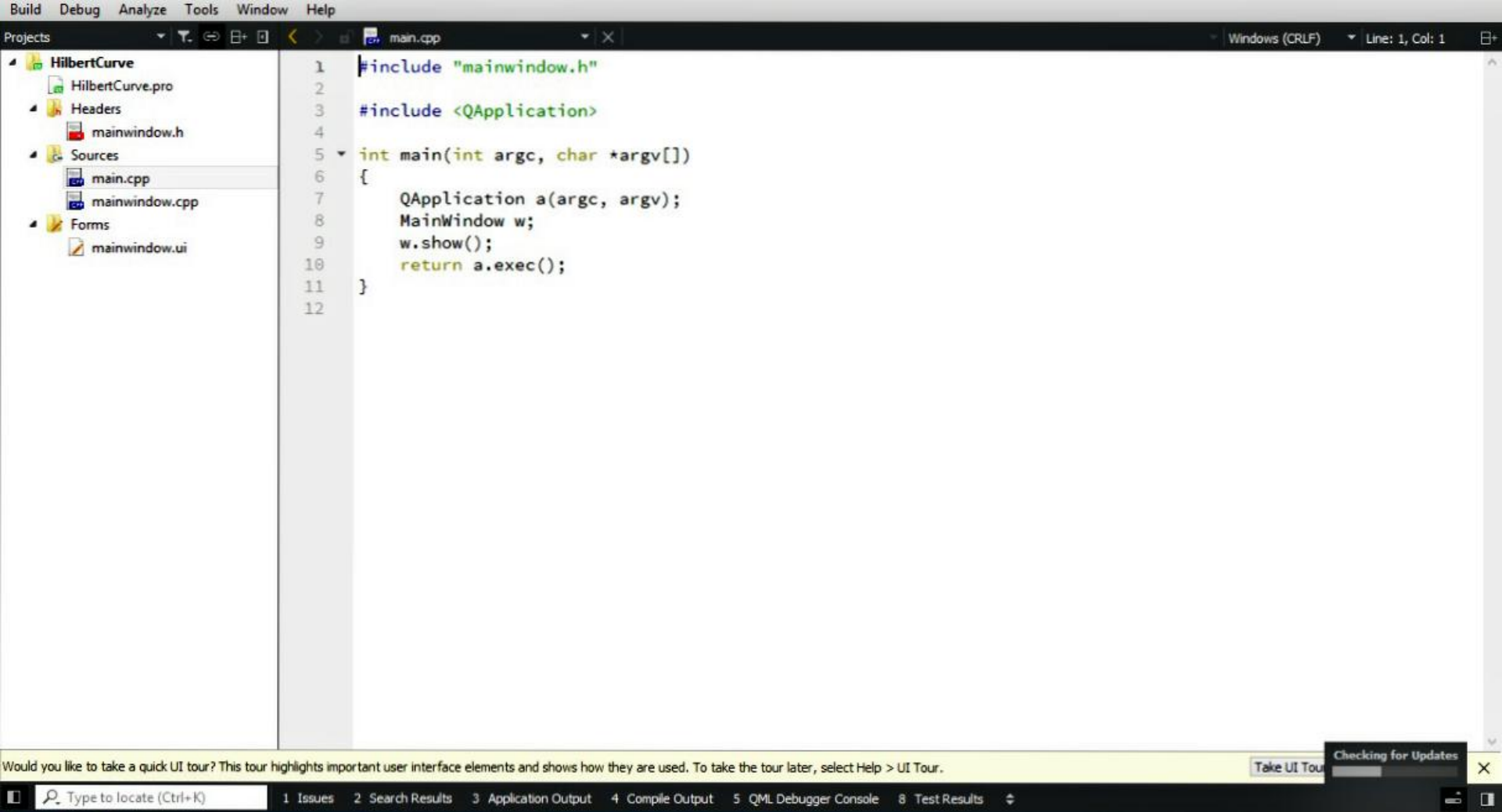
ii) Order - 2

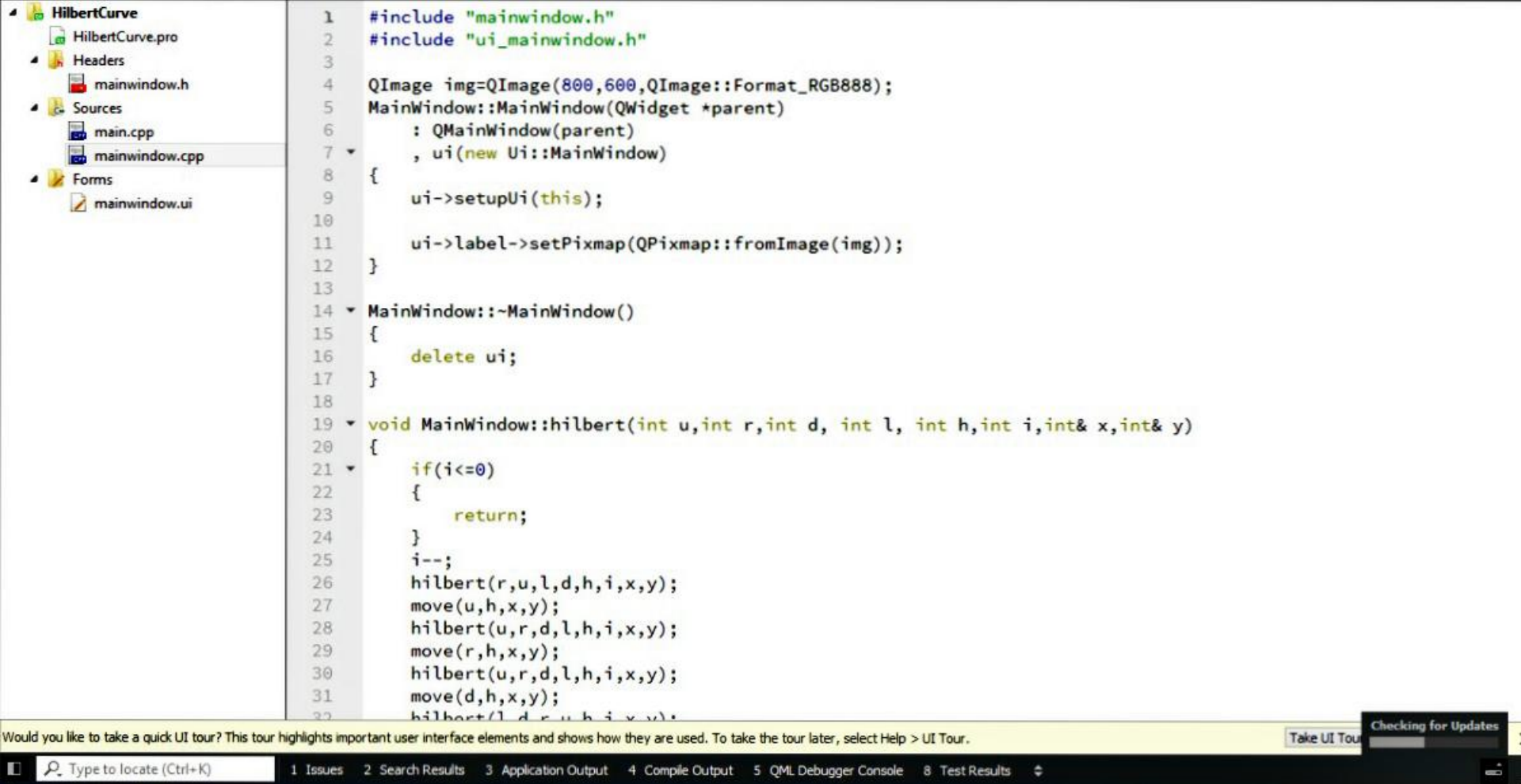


No. *	Order	x_1	y_1	x_2	y_2
1.	1	25	75	25	35
		25	35	65	35
		65	35	65	75

Conclusion :-
Succefully implemented the concept of Hilbert curve to make a pattern on QT creator.

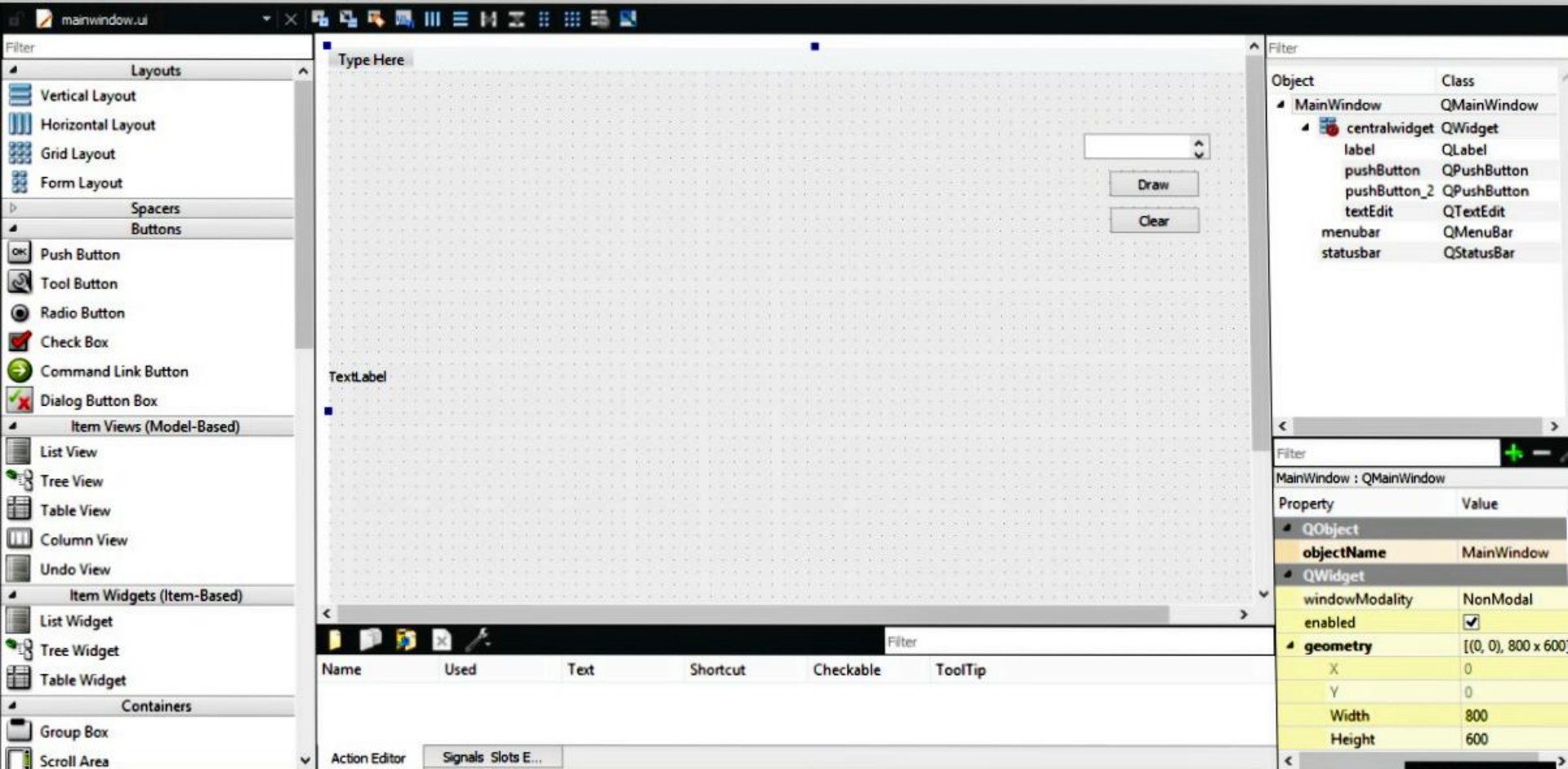

Build Debug Analyze Tools Window Help
Projects
mainwindow.h
<Select Symbol>
Windows (CRLF)
Line: 1, Col: 1
HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
#ifndef MAINWINDOW_H
#define MAINWINDOW_H
#include <QMainWindow>
QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE
class MainWindow : public QMainWindow
{
Q_OBJECT
public:
MainWindow(QWidget *parent = nullptr);
~MainWindow();
int n;
void hilbert(int u,int r,int d, int l, int h,int i,int& x,int& y);
void dda(float x1, float y1, float x2, float y2);
void move(int j, int h, int&x, int &y);
private slots:
void on_pushButton_clicked();
void on_pushButton_2_clicked();
private:
Ui::MainWindow *ui;
};
#endif // MAINWINDOW_H
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tour
Checking for Updates
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML Debugger Console
8 Test Results

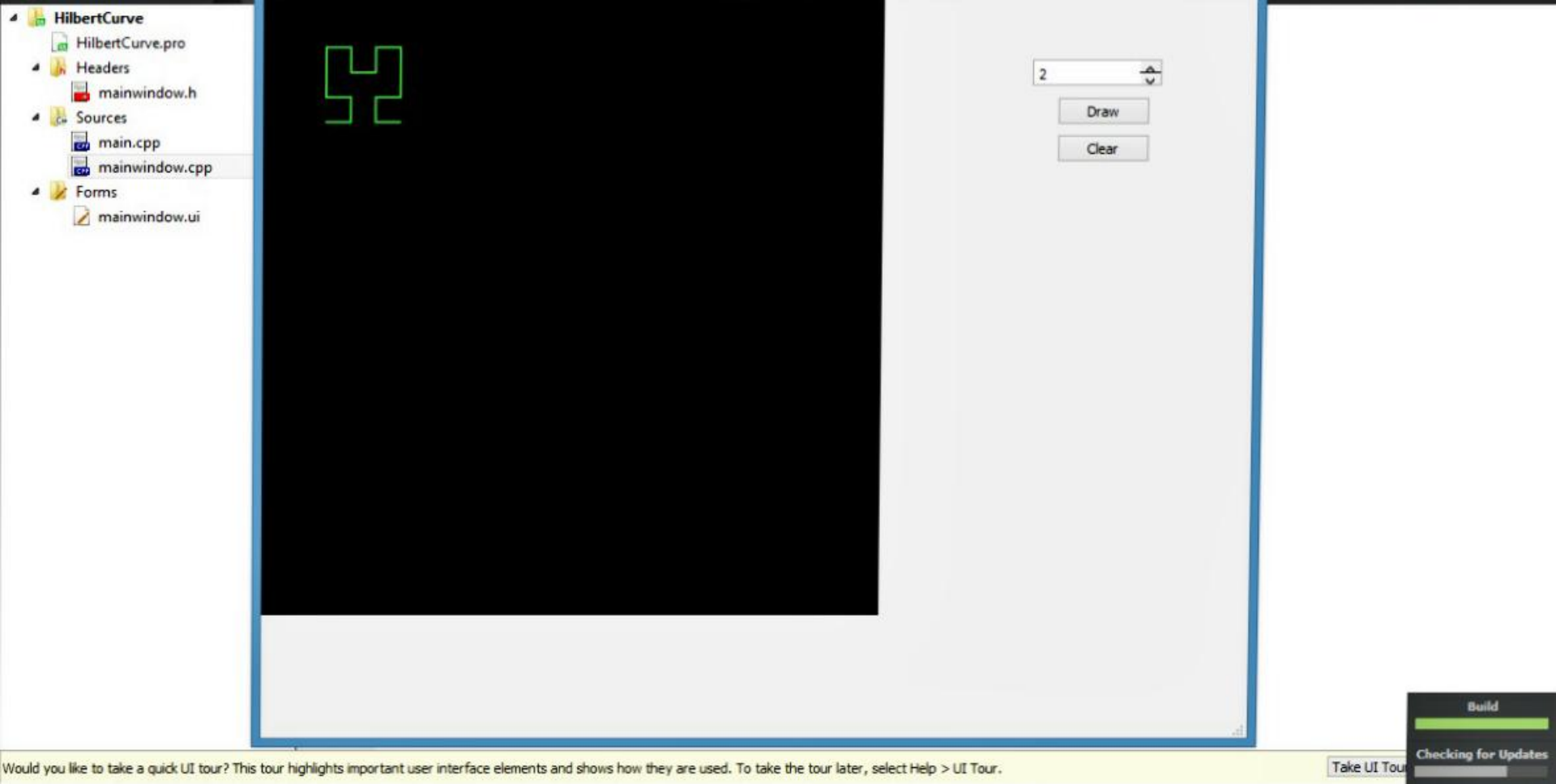

Build Debug Analyze Tools Window Help
Projects
main.cpp
Windows (CRLF)
Line: 1, Col: 1
HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
#include "mainwindow.h"
#include <QApplication>
int main(int argc, char *argv[])
{
QApplication a(argc, argv);
MainWindow w;
w.show();
return a.exec();
}
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tour
Checking for Updates
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML Debugger Console
8 Test Results


HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
#include "mainwindow.h"
#include "ui_mainwindow.h"
QImage img=QImage(800,600,QImage::Format_RGB888);
MainWindow::MainWindow(QWidget *parent)
: QMainWindow(parent)
, ui(new Ui::MainWindow)
{
ui->setupUi(this);
ui->label->setPixmap(QPixmap::fromImage(img));
}
MainWindow::~MainWindow()
{
delete ui;
}
void MainWindow::hilbert(int u,int r,int d, int l, int h,int i,int& x,int& y)
{
if(i<=0)
{
return;
}
i--;
hilbert(r,u,l,d,h,i,x,y);
move(u,h,x,y);
hilbert(u,r,d,l,h,i,x,y);
move(r,h,x,y);
hilbert(u,r,d,l,h,i,x,y);
move(d,h,x,y);
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tour
Checking for Updates
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML Debugger Console
8 Test Results


HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
hilbert(l,d,r,u,h,i,x,y);
}
void MainWindow::move(int j,int h,int& x,int& y)
{
int x1=x,y1=y;
switch(j)
{
case 1:
y-=h; //up
break;
case 2:
x+=h; //right
break;
case 3 :
y+=h; //down
break;
case 4:
x-=h; //left
break;
}
dda(x1,y1,x,y);
}
void MainWindow::dda(float x1,float y1,float x2,float y2)
{
QRgb val=qRgb(0,255,0);
float dx=x2-x1,dy=y2-y1,x=x1,y=y1;
float steps=abs(dx)>abs(dy)?abs(dx):abs(dy);
float xinc=dx/steps,yinc=dy/steps;
for(int i=0;i<=steps;i++)
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tour
Checking for Updates
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML Debugger Console
8 Test Results


HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
56
57 void MainWindow::dda(float x1,float y1,float x2,float y2)
58 {
59 QRgb val=qRgb(0,255,0);
60 float dx=x2-x1,dy=y2-y1,x=x1,y=y1;
61 float steps=abs(dx)>abs(dy)?abs(dx):abs(dy);
62 float xinc=dx/steps,yinc=dy/steps;
63 for(int i=0;i<=steps;i++)
64 {
65 img.setPixel(x,y,val);
66 x+=xinc;
67 y+=yinc;
68 }
69 ui->label->setPixmap(QPixmap::fromImage(img));
70 }
71
72 void MainWindow::on_pushButton_clicked()
73 {
74 n = ui->textEdit->toPlainText().toInt();
75 int x=n*25,y=n*75;
76 hilbert(1,2,3,4,40/n,n,x,y);
77 }
78
79
80 void MainWindow::on_pushButton_2_clicked()
81 {
82 //n=0;
83 img.fill(0);
84 ui->label->setPixmap(QPixmap::fromImage(img));
85 }
86
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tou
Checking for Updates
Type to locate (Ctrl+K)
1 Issues
2 Search Results
3 Application Output
4 Compile Output
5 QML Debugger Console
8 Test Results


mainwindow.ui
Filter
Layouts
Vertical Layout
Horizontal Layout
Grid Layout
Form Layout
Spacers
Buttons
Push Button
Tool Button
Radio Button
Check Box
Command Link Button
Dialog Button Box
Item Views (Model-Based)
List View
Tree View
Table View
Column View
Undo View
Item Widgets (Item-Based)
List Widget
Tree Widget
Table Widget
Containers
Group Box
Scroll Area
Type Here
Draw
Clear
TextLabel
Filter
Name
Used
Text
Shortcut
Checkable
ToolTip
Action Editor
Signals Slots E...
Filter
Object
Class
MainWindow
QMainWindow
centralwidget
QWidget
label
QLabel
pushButton
QPushButton
pushButton_2
QPushButton
textEdit
QTextEdit
menubar
QMenuBar
statusbar
QStatusBar
Filter
MainWindow : QMainWindow
Property
Value
QObject
objectName
MainWindow
QWidget
windowModality
NonModal
enabled
geometry
[(0, 0), 800 x 600]
X
0
Y
0
Width
800
Height
600


HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
2
Draw
Clear
Build
Checking for Updates
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.
Take UI Tour


HilbertCurve
HilbertCurve.pro
Headers
mainwindow.h
Sources
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
5
Draw
Clear
Would you like to take a quick UI tour? This tour highlights important user interface elements and shows how they are used. To take the tour later, select Help > UI Tour.