---------------------------------------------------------------------- Write C++/Java program to generate Hilbert curve using concept of fractals

----------------------------------------------------------------------#include "hilbs.h"

#include "ui\_hilbs.h"

#include<math.h>

#include<QPainter>

#include<QScreen>

#include<QColor>

hilbs::hilbs(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::hilbs)

{

ui->setupUi(this);

}

hilbs::~hilbs()

{

delete ui;

}

void hilbs::changeEvent(QEvent \*e)

{

QMainWindow::changeEvent(e);

switch (e->type()) {

case QEvent::LanguageChange:

ui->retranslateUi(this);

break;

default:

break;

}

}

void hilbs :: paintEvent(QPaintEvent \*event)

{

int x,y,r=2,d=3,l=4,u=1,n,h=10,x0,y0;

n=ui->N->toPlainText().toInt();

QScreen \*screen = QGuiApplication::primaryScreen();

QRect screenGeometry = screen->geometry();

int height = screenGeometry.height();//1360;//

int width = screenGeometry.width();

p1=x=width/4;p2=y=height/4;

//pr=50,pg=47,pb=80;

hilbert(r,d,l,u,n,h,x,y);

}

void hilbs :: move(int j, int h, int &x, int &y)

{

//PainterPath path;

QPainter pain(this);

pain.setPen(Qt::black);

// pain.setPen(QColor(pr,pg,pb,127));

p1=x,p2=y;

if(j==1)

y-=h;

else if(j==2)

x+=h;

else if(j==3)

y+=h;

else if(j==4)

x-=h;

pain.drawLine(p1,p2,x,y);

//p.drawLine(100,100,500,500);//p.drawPath(path);

}

void hilbs :: hilbert(int r, int d, int l, int u, int i, int h, int &x, int &y)

{

if(i>0)

{

i--;

hilbert(d,r,u,l,i,h,x,y);

//pr+=2;

move(r,h,x,y);

hilbert(r,d,l,u,i,h,x,y);

//pg+=2;

move(d,h,x,y);

hilbert(r,d,l,u,i,h,x,y);

//pb+=2;

move(l,h,x,y);

hilbert(u,l,d,r,i,h,x,y);

}

}

void hilbs::on\_Draw\_clicked()

{

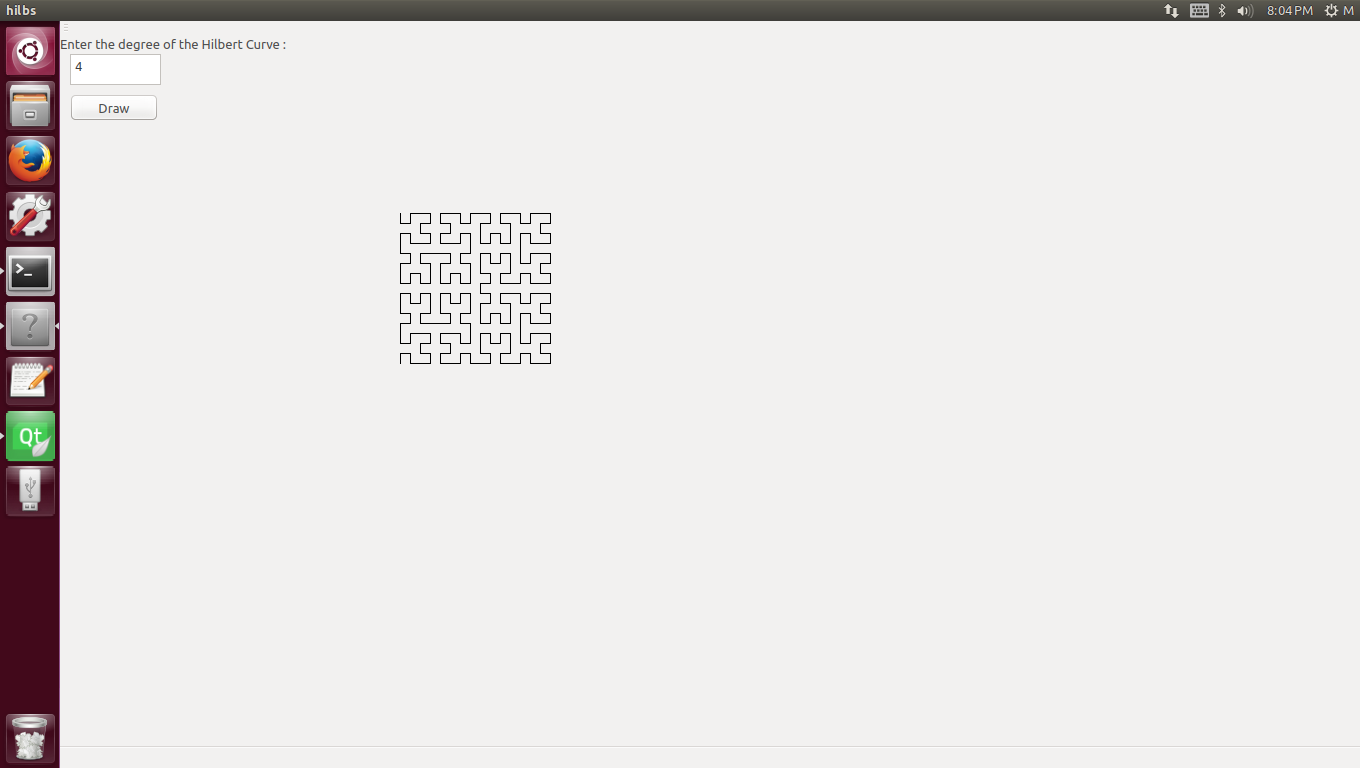
flag=0;

update();

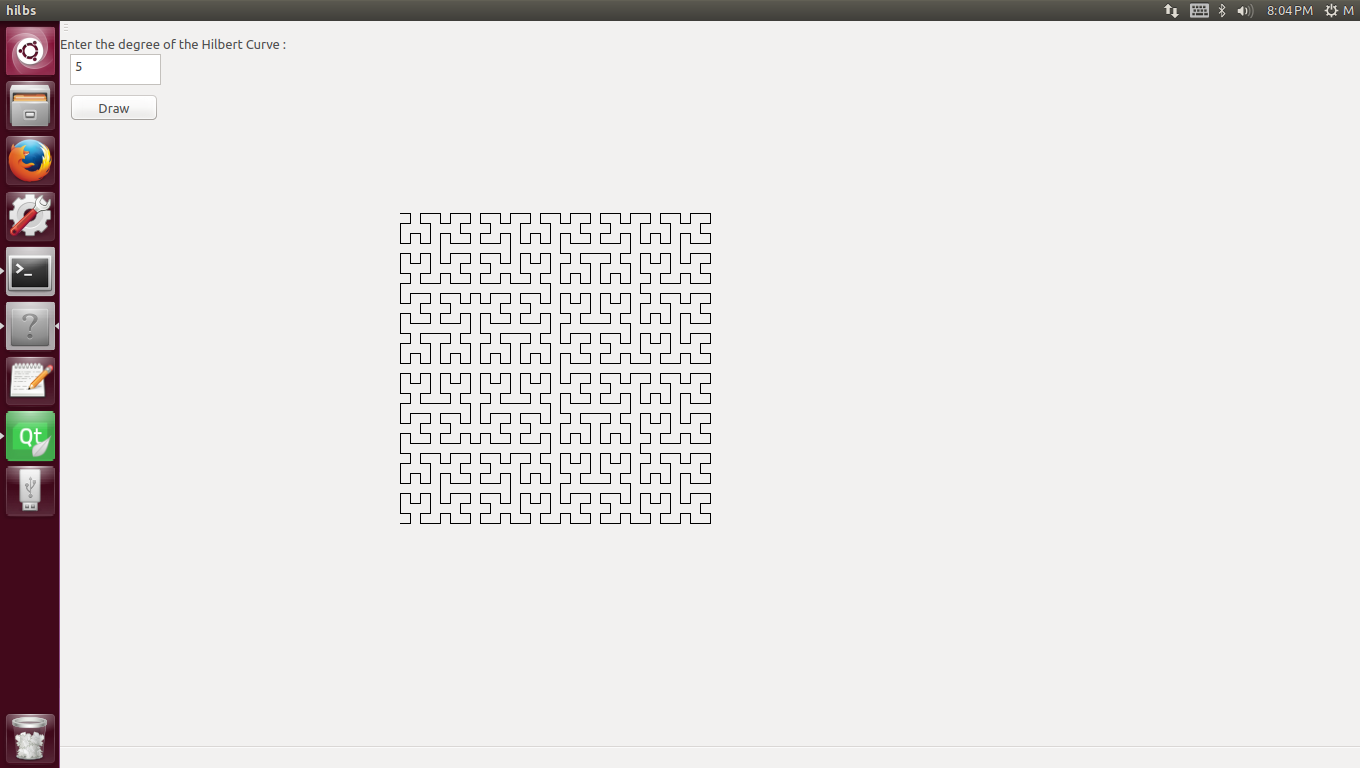
}

**OUTPUT:**

**Hilberts curve for n=4**

****

**Hilberts curve for n=5**

**** for n= 4