****

第二次实验报告

2020年6月5日

题目：二维图形类

专业:18软件工程

班级:软件一班

姓名:严灏

学号:201830020446

1、题目描述

设计一个图形类结构,以Shape为基类,包含各种常见二维图形:圆形、多边形、三角形、四边形、平行四边形、菱形、矩形、正方形等

  程序实现的功能:

1.      可创建图形的对象,最好能结合运算符重载实现输入/输出等各种功能

2.      可求图形的周长和面积

3.      可通过二维坐标来确定图形位置和大小(\*是否可旋转选做)

2先给出所有代码，然后是输出

Main.cpp

#include "mainwindow.h"

#include <QApplication>

#include <iostream>

#include <QDebug>

#include <QtDebug>

#include <iomanip>

using namespace std;

int **main**(int argc, char \*argv[])

{

QApplication a(*argc*, argv);

MainWindow w;

w.show();

return a.exec();

}

Mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QPainter>

#include <QRect>

#include <QBrush>

#include <QPen>

#include <string>

#include <QFont>

#include <qstring.h>

using std::string;

QT\_BEGIN\_NAMESPACE

namespace **Ui** { class **MainWindow**; }

QT\_END\_NAMESPACE

class **MainWindow** : public QMainWindow

{

Q\_OBJECT

public:

**MainWindow**(QWidget \*parent = nullptr);

~***MainWindow***();

void ***paintEvent***(QPaintEvent \*event);

void **calculate**();

private:

Ui::MainWindow \*ui;

};

#endif // MAINWINDOW\_H

Mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "Shape.h"

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(parent)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

calculate();

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::***paintEvent***(QPaintEvent \*event)

{

QPainter painter(this); // 创建QPainter一个对象

QPen pen;

//画圆

painter.drawEllipse(QPoint(width()/2,height()/2),100,100);

// 画一个四边形

QPolygonF polygon;

polygon << QPointF(350.0, 20.0) << QPointF(430.0, 30.0) << QPointF(430.0, 80.0) << QPointF(370.0, 90.0);

painter.drawPolygon(polygon, Qt::WindingFill);

// 画一个空心正方形

pen.setColor(Qt::black);

painter.setPen(pen);

painter.drawRect(QRect(1, 1, 100, 100));

// 画一个实心正方形

QBrush bruch(Qt::green); // 画刷

painter.setBrush(bruch); // 设置画刷

painter.drawRect(QRect(110, 1, 200, 100));

// 画一个多边形

QPolygonF polygon2;

polygon2 << QPointF(200.0, 320.0) << QPointF(230.0, 230.0) << QPointF(300.0, 320.0) << QPointF(260.0, 400.0)<< QPointF(200.0, 400.0);

painter.drawPolygon(polygon2, Qt::WindingFill);

}

void MainWindow::**calculate**()

{

Square square(1,1,200);

string a=u8"正方形的面积是: "+std::to\_string(square.*getArea*());

Quadrangle aa(10,10,20,20,30,30,40,40);

string b=u8"无规则四边形的周长是"+std::to\_string(aa.*getPerimeter*());

string c=u8"无规则四边形的面积是"+std::to\_string(aa.*getArea*() );

Circle ci(1,2,3);

string d=u8"圆的面积是:"+std::to\_string(ci.*getArea*());

string e=u8"圆的周长是:"+std::to\_string(ci.*getPerimeter*());

string temp=a+'\n'+b+'\n'+c+'\n'+d+'\n'+e;

ui->textEdit->setText(QString::fromStdString(temp));

}

Shape.h

#ifndef SHAPE\_H

#define SHAPE\_H

#include <iostream>

#include <QMainWindow>

#include <QPainter>

#include <QRect>

#include <QBrush>

#include <QPen>

#include <QFont>

#include <vector>

#include <iostream>

using std::cout;

using std::vector;

using std::istream;

#define PI 3.1415926

struct **point**//保存点和向量

{

int x, y;

**point**(int X=0,int Y=0):x(X),y(Y){}

point& operator=(point p) {

x = p.x; y = p.y;

return \*this;

}

double **length**() {//求长度

return sqrt(x \* x + y \* y);

}

int **len2**() {//求长度平方

return x \* x + y \* y;

}

};

int operator\*(point a, point b)//叉积

{

return a.x \* b.y - a.y \* b.x;

}

int operator^(point a, point b)//点积

{

return a.x \* b.x + a.y \* b.y;

}

point operator+(point a, point b)

{

return point(a.x + b.x, a.y + b.y);

}

point operator-(point a, point b)

{

return point(a.x - b.x, a.y - b.y);

}

class **Shape**

{

public:

virtual double ***getPerimeter***(){return 0;} ;

virtual double ***getArea***(){return 0;} ;

};

//圆

class **Circle**:public Shape

{

public:

explicit **Circle**(double x,double y,double radius):radius(radius)

{

center[0] = x;

center[1] = y;

}

virtual double ***getArea***()

{

m\_area = PI \* radius \* radius;

return m\_area;

}

virtual double ***getPerimeter***()

{

m\_perimeter = 2 \* PI \* radius;

return m\_perimeter;

}

virtual void ***print***()

{

}

virtual bool ***symmetrical***()

{

return true;

}

private:

double center[2];

double radius;

double m\_perimeter;

double m\_area;

};

//三角形

class **Triangle**:public Shape

{

};

//四边形

class **Quadrangle**:public Shape

{

protected:

point p1, p2, p3, p4;

double l1, l2, l3, l4;

point c1, c2, c3, c4;//保存4条边向量

int area;//面积

double peri;//周长

public:

friend istream& operator>>(istream& c, Quadrangle& x);

double ***getArea***();

double ***getPerimeter***();

void **setPoint**(int val[8]);

//更新属性,求面积和周长

void **updateProperty**();

//默认构造函数

**Quadrangle**(int x1 = 0, int y1 = 0, int x2 = 0, int y2 = 0, int x3 = 0, int y3 = 0, int x4 = 0, int y4 = 0);

//赋值构造函数

**Quadrangle**(Quadrangle& px);

Quadrangle& operator=(Quadrangle px);

public:

vector<int> loc;

};

inline double Quadrangle::***getArea***() { return area; }

inline double Quadrangle::***getPerimeter***() { return peri; }

inline void Quadrangle::**setPoint**(int val[8]) {

p1 = point(val[0], val[1]);

p2 = point(val[2], val[3]);

p3 = point(val[4], val[5]);

p4 = point(val[6], val[7]);

updateProperty();

}

//更新属性,求面积和周长

inline void Quadrangle::**updateProperty**() {

//以p1为中心点依次求叉积解出四边形面积

c1 = p2 - p1;

c2 = p3 - p1;

c3 = p4 - p1;

area = abs(c1 \* c2 + c2 \* c3);

//然后依次保存4边向量并求出周长

c2 = p3 - p2;

c3 = p4 - p3;

c4 = p1 - p4;

l1 = c1.length();

l2 = c2.length();

l3 = c3.length();

l4 = c4.length();

peri = l1 + l2 + l3 + l4;

}

//默认构造函数

inline Quadrangle::**Quadrangle**(int x1, int y1, int x2, int y2, int x3, int y3, int x4, int y4) {

int val[] = { x1,y1,x2,y2,x3,y3,x4,y4 };

setPoint(val);

}

//赋值构造函数

inline Quadrangle::**Quadrangle**(Quadrangle& px) {

p1 = px.p1; p2 = px.p2; p3 = px.p3; p4 = px.p4;

updateProperty();

}

//赋值运算符

Quadrangle& Quadrangle::operator=(Quadrangle px)

{

p1 = px.p1; p2 = px.p2; p3 = px.p3; p4 = px.p4;

updateProperty();

return \*this;

}

//多边形

class **Polygon**:public Shape

{

public:

};

//菱形

class **Rhombus**:public Quadrangle

{

public:

};

//平行四边形

class **Parallelogram** :public Quadrangle

{

public:

**Parallelogram**(Quadrangle px);

//平行四边形判断:两组对边平行

virtual bool ***check***();

//构造函数

};

inline Parallelogram::**Parallelogram**(Quadrangle px) :Quadrangle(*px*) {

}

inline bool Parallelogram::***check***() {

return (c1 \* c3 == 0) && (c2 \* c4 == 0);

return false;

}

//正方形

class **Square**:public Quadrangle

{

public:

**Square**(int x,int y,int sideLength):Quadrangle(x,y,x+sideLength,y,x+sideLength,y+sideLength,x,y+sideLength)

{

}

//构造函数

**Square**(Quadrangle px);

};

//矩形

class **Rectangele**:public Quadrangle

{

public:

**Rectangele**(Quadrangle px);

**Rectangele**(int x,int y,int width,int height):Quadrangle(x,y,x+width,y,x+width,y+height,x,y+height)

{

}

virtual bool ***check***();

//构造函数

};

inline Rectangele::**Rectangele**(Quadrangle px) :Quadrangle(*px*) {

}

inline bool Rectangele::***check***() {

Parallelogram p(\*this);

if (!p.*check*()) return false;

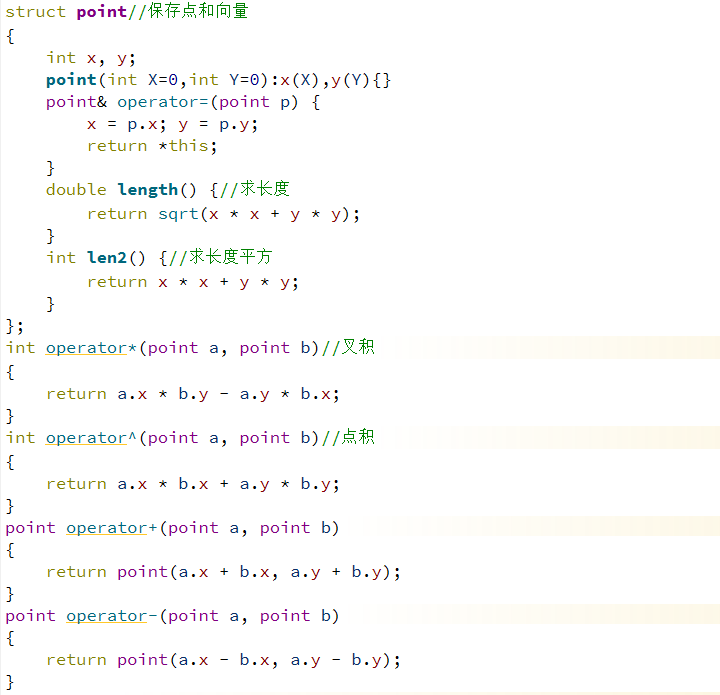
return ((c1 ^ c2) == 0);

}

#endif // SHAPE\_H

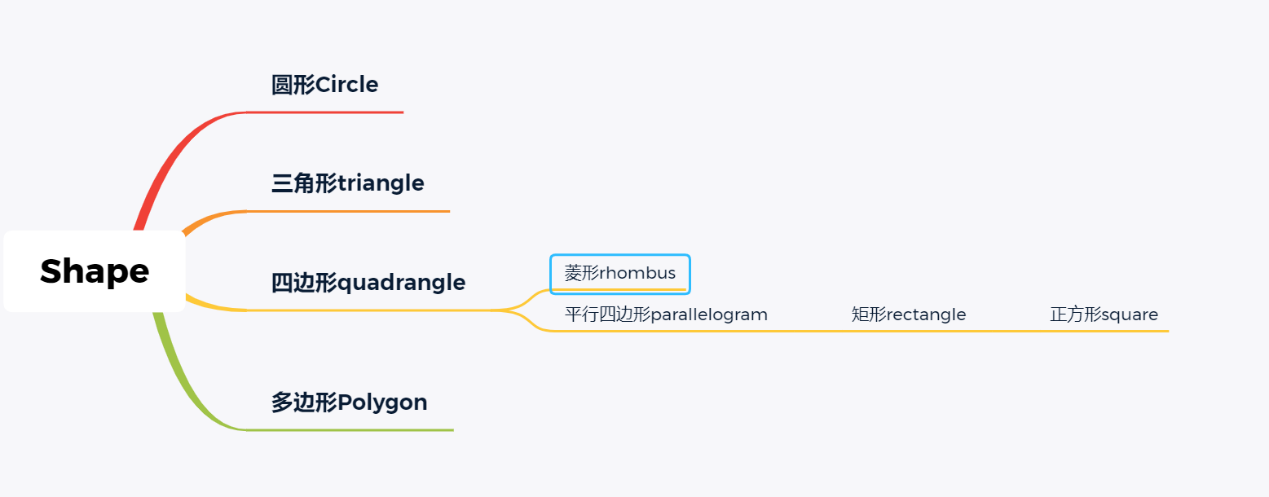
3.实验截图:

**1.重载点积等运算符**

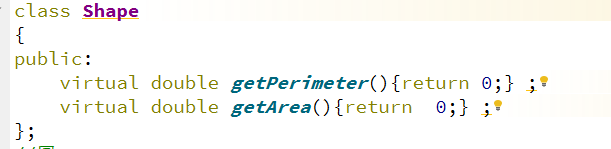


**2.类的层次结构**

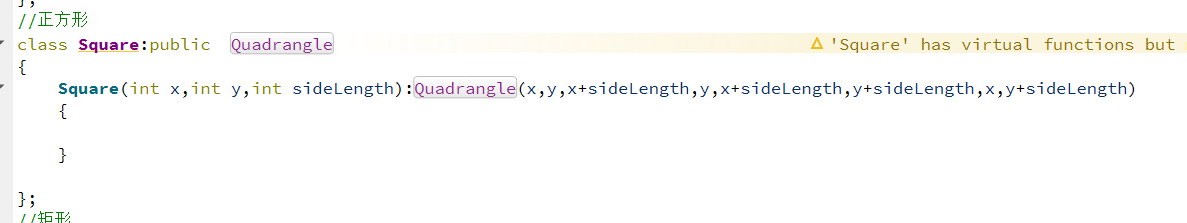
**利用Xmind绘制类的层次结构**



**3.虚函数**



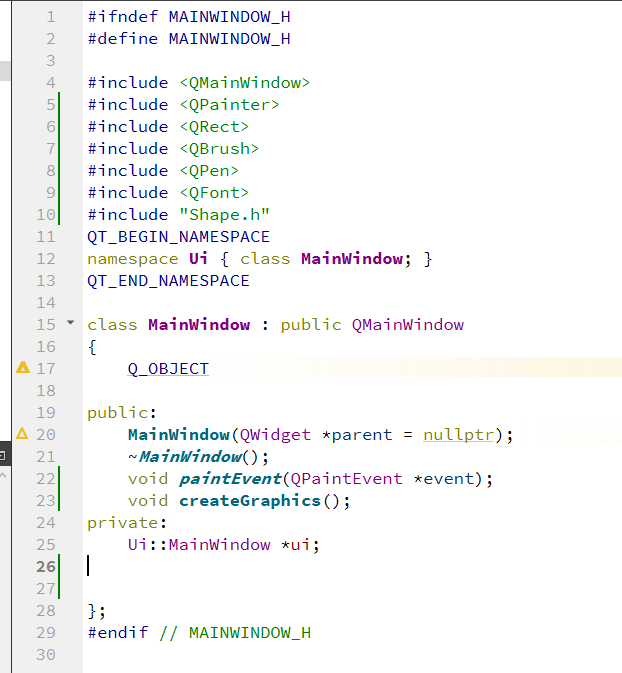
**4.调用父类的构造函数**



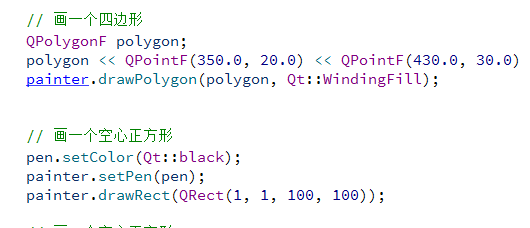
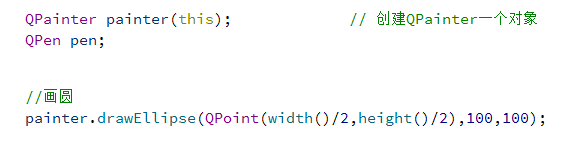
**5.重写父类的虚函数**

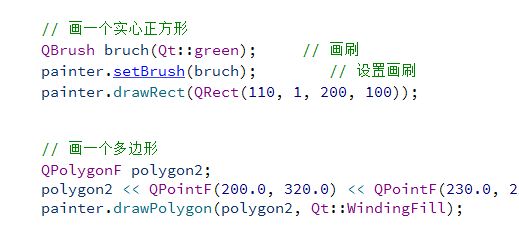


6.QT中负责画图的类



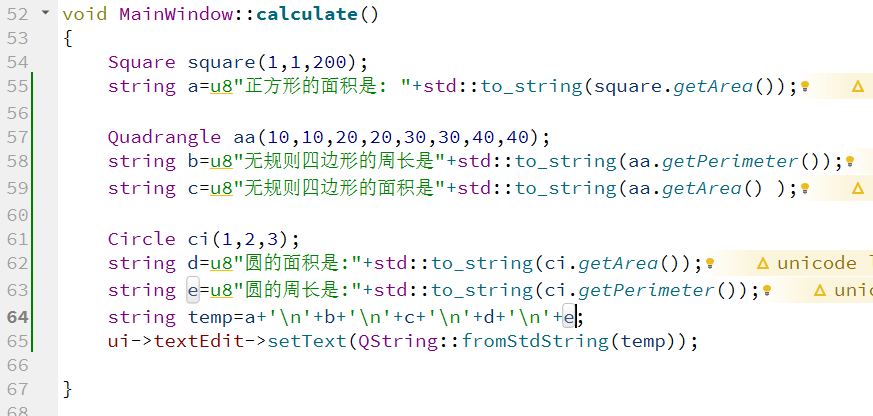
**7.画圆的代码**





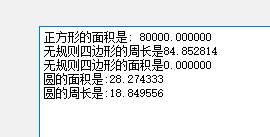
**8最终效果**

**分别画了正方形, 矩形, 四边形, 多边形 和圆**



**9.其中,显示面积 周长部分**

**测试代码mainwindow.cpp里面**



**4、完成度分析**

要分析自己实现的功能与给定功能之间的差别，是否超出给定的功能

基本要求:

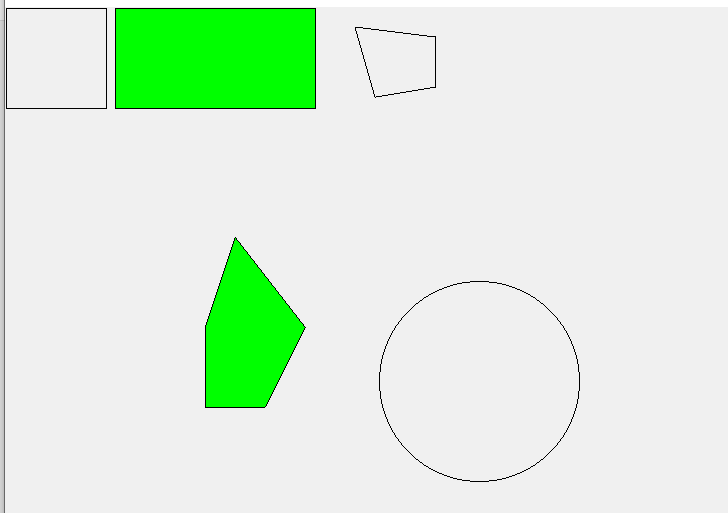
1.      可创建图形的对象,最好能结合运算符重载实现输入/输出等各种功能(完成)

2.      可求图形的周长和面积(完成)

3.      可通过二维坐标来确定图形位置和大小(\*是否可旋转选做)(完成)

额外完成:  
1.利用Qt提供的Qpainter类进行绘图 把正方形,矩形,不规则四边形,

五边形等多边形,圆等绘制出来了



5.实验总结

通过这次的实验,我对类的层次结构有了更好理解,同时对C++如何通过虚函数来实现多态有了更深的理解. 在完成基本任务的同时

我利用自己对GUI绘图的了解,完成了二维图形的绘图,收获很大.