C++程序设计报告

姓名：

学院：

班级：

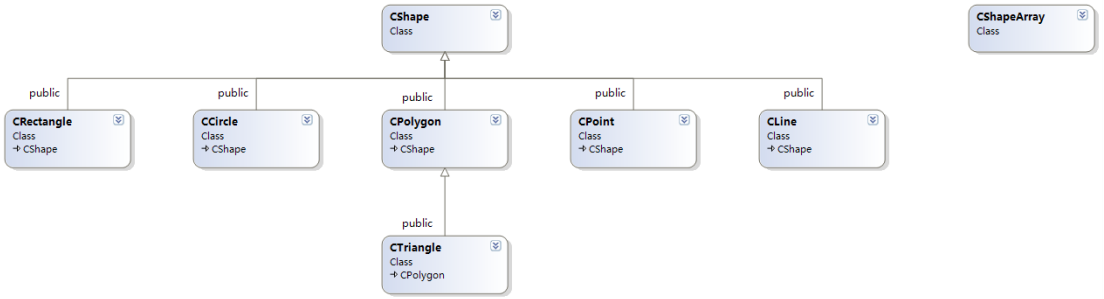
学号：

一、设计目的：

实现一个平面上的形状编辑程序。

二、程序功能简介：

1. 按照下图给出的层次关系来定义类。



2. 所有形状支持无参数构造，有参数构造，拷贝构造，析构。
3. 所有形状支持平移操作，需要重载 operator+。
4. 所有形状（除去无意义的），均支持计算周长。
5. 所有形状（除去无意义的），均支持 Draw（）操作，此时只要显示形状的名称，位置等信息。
6. 需要实现一个 CShapeArray 类，该类类似一个数组，用来存放编辑过程中的平面形状。该类需要支持：添加，插入，删除，查询，复制等操作。可以支持形状编辑中需要的针对形状的操作。
7. 主程序中实现用户输入形状及其参数，然后把形状存入 6 中定义的 CShapeArray。在输入形状的同时，用户可以查询当前已经输入的形状（可按名称（需要对每个平面形状加入名称），位置来查询）。支持用户对形状的复制，粘贴（粘贴时假设用户指定粘贴的位置）。同时支持用户对形状的删除操作。
8. 输入和处理好的形状可以存入文件，并从文件中读入。
9. 支持对当前所有形状的 Draw()。

三、详细介绍：

1、关于程序使用的代码风格

本程序使用 Visual Studio 2010 编写，并按照规范风格，将类、函数、变量的声明、其他头文件的引用等写在头文件(*.h 文件)中，并使用头文件保护符（#pragma once）保护以免重复编译，具体函数的定义等则写在对应的源文件中。函数的注释说明写在头文件函数声明处，函数内部具体代码说明写在源文件代码处。

2、程序中类的声明

本程序定义了以下几个类(**其中由于 CPoint 为 VS 中的保留类名，改用 CCPoint**)：

```
class CShape;
class CCircle :        public CShape;
class CCPoint :        public CShape;
class CLine :          public CShape;
class CRectangle :     public CShape;
class CPolygon :       public CShape;
class CTriangle :      public CPolygon;
class CShapeArray;
```

基类 CShape 定义如下，其中函数

```
    virtual void Draw(void);
    virtual void Calc(void);
    virtual void saveToFile(ofstream&);
    virtual CShape& loadFromFile(ifstream&);
    virtual int exist();
```

为虚函数，具体根据继承的子类不同而有不同定义，体现了 C++的多态性；
变量 const static int SHAPE_CIRCLE 等用于区分不同子类的具体类型，用于全局函数 loadFromFile()从文件加载形状信息。

```
class CShape
{
public:
    const static int SHAPE_CIRCLE = 1;
    const static int SHAPE_POINT = 2;
    const static int SHAPE_LINE = 3;
    const static int SHAPE_POLYGON = 4;
    const static int SHAPE_RECTANGLE = 5;
    const static int SHAPE_TRIANGLE = 6;
```

```
    // 无参数构造
    CShape(void);
    // 有参数构造
    CShape(char*);
    // 析构函数
    ~CShape(void);
    // 拷贝构造
    CShape(const CShape&);
    // 获取名称
    char *getName();
    // 显示名称等信息，使用虚函数
    virtual void Draw(void);
    // 计算并显示周长，使用虚函数
    virtual void Calc(void);
    // 保存至文件
    virtual void saveToFile(ofstream&);
    // 从文件读取
    virtual CShape& loadFromFile(ifstream&);
    // 判断形状是否存在
    virtual int exist();

protected:
    // 图形的名称
    char* name;
    int shape;
};
```

子类 CCircle, CCPoint, CLine, CRectangle, CPolygon 继承于基类 CShape，类 CTriangle 继承于类 CPolygon，分别表示具体的形状，并分别定义了各自的 Draw()，Calc()，saveToFile， loadFromFile 等函数。

类 CShapeArray 用于存放编辑过程中的平面形状，支持：添加，插入，删除，查询，复制等操作。使用 vector<CShape*> vec 来保存添加进的 CShape 对象的指针。其定义如下：

```
class CShapeArray
{
public:
    CShapeArray(void);
    ~CShapeArray(void);
    // 添加
    void add(CShape*);
```

```
    // 插入
    void insert(int, CShape);
    // 删除
    void del(int);
    // 清除全部
    void clear(void);
    // 查询
    CShape* get(int);
    // 复制
    void copy(int, int);
    // 显示所有元素
    void drawAll(void);
    // 获取元素数目
    int getSize(void);
    // 根据名称查询
    int findByName(char*);

private:
    vector<CShape*> vec;
};
```

3、全局变量及全局函数

```
/**********************全局变量声明**********************/
// 保存到的文件名
const static char *fname = "D:\\dat.txt";
// CShapeArray 对象
CShapeArray arr;
// 记录要复制的形状所在位置
int copyPos = -1;
/************************函数声明************************/
// 输入数字进行选择
int input(char *, int);
// 清屏
void clr();
// 暂停
void pause();
// 选择操作
void inputOp();
// 选择形状
void inputShape();
// 查询
void inputQuery();
```

```
// 输入位置查询
void inputPos();
// 输入名称查询
void inputName();
// 复制
void inputCopy();
// 粘贴
void inputPaste();
// 删除
void inputDel();
// 保存至文件
void saveToFile();
// 从文件读取
void loadFromFile();
// 输入矩形
CRectangle* inputRectangle();
// 输入圆形
CCircle* inputCircle();
// 输入多边形
CPolygon* inputPolygon();
// 输入三角形
CTriangle* inputTriangle();
// 输入点
CCPoint* inputPoint();
// 输入直线
CLine* inputLine();
```

4、程序的使用

程序启动即进入功能选择主菜单。用户通过输入数字并按回车进行功能的选择，如果用户输入的不是数字，或数字超出了输入范围，则提示输入错误需要重新输入(该功能由 input 函数实现)。

添加形状时，根据用户选择的形状不同，要求输入的参数也不同，如果输入的参数能构成相应形状（如三角形两边之和要大于第三边，此功能由 exist 函数实现），则添加该形状至对象 CShapeArray arr 中，否则提示错误而不保存。

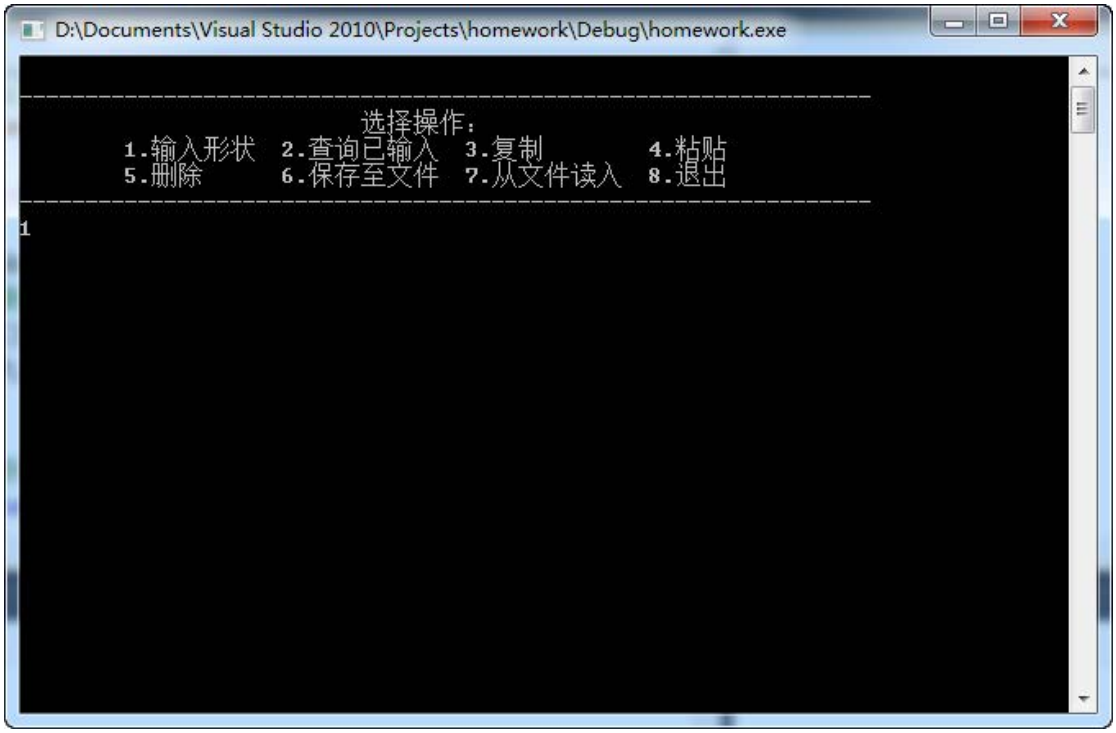
输入形状的名称时，支持输入中英文，并可包含空格。

5、程序的稳定性

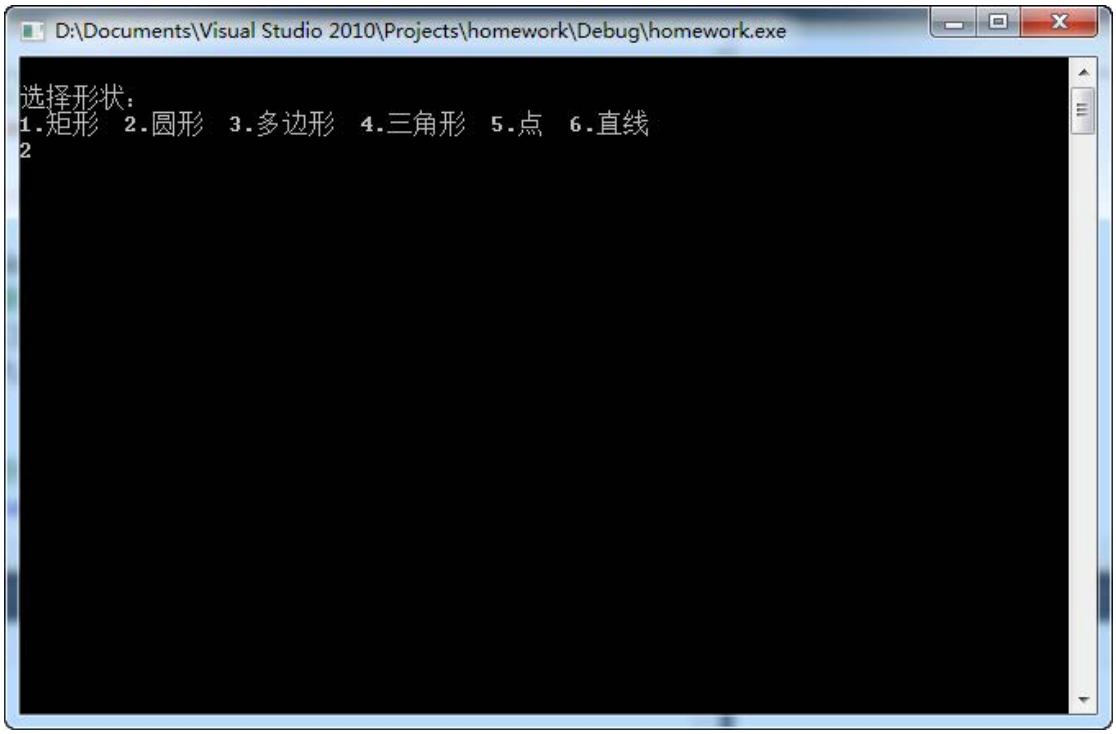
本程序能自动纠正部分用户输入错误，以及从文件读取时发生的错误，并给出相应提示信息。

6、运行结果

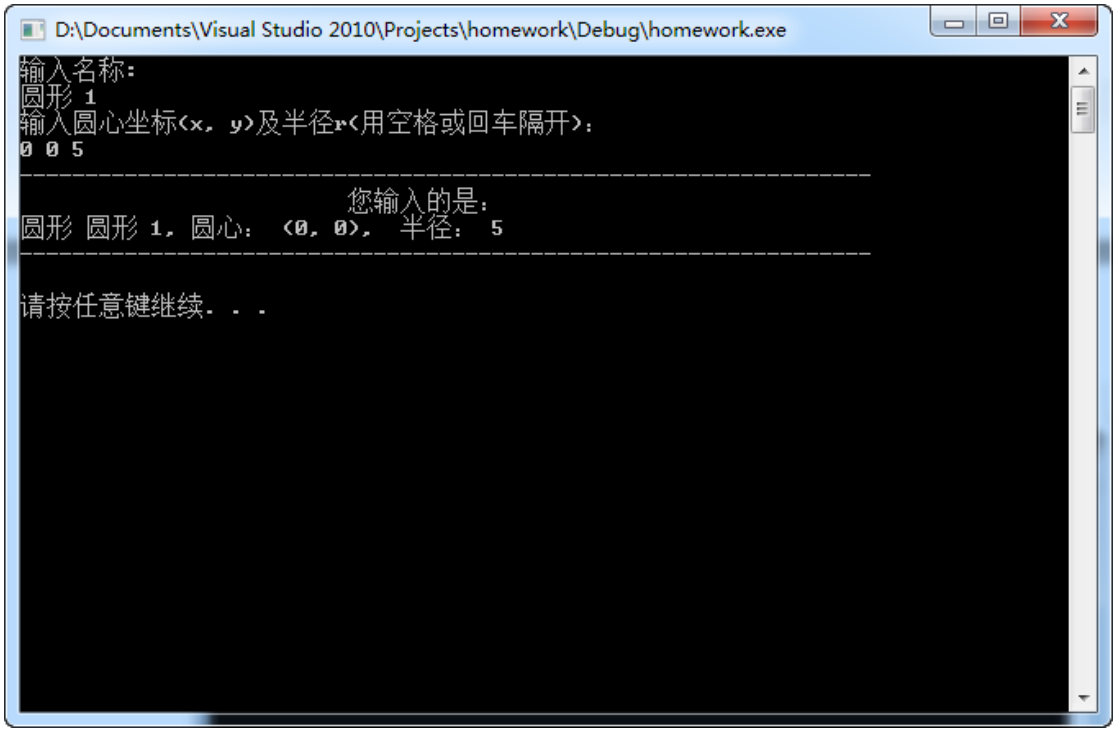
启动后提示输入数字选择操作：



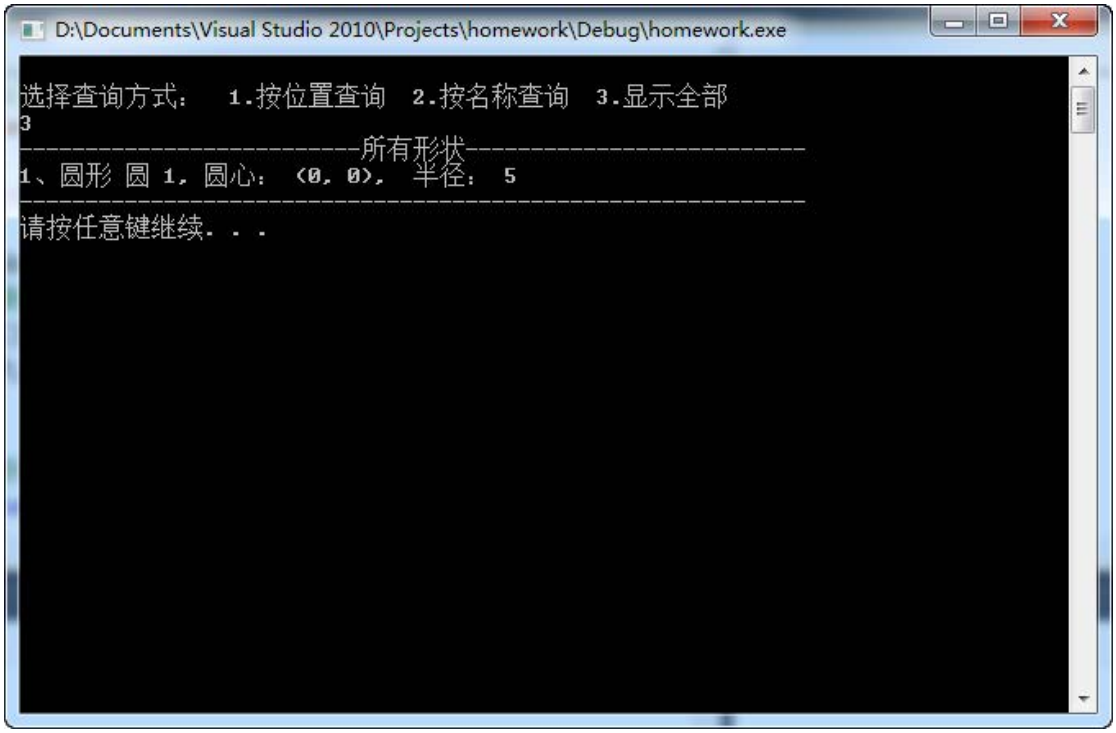
选择操作输入形状，选择要输入的形状为圆形：



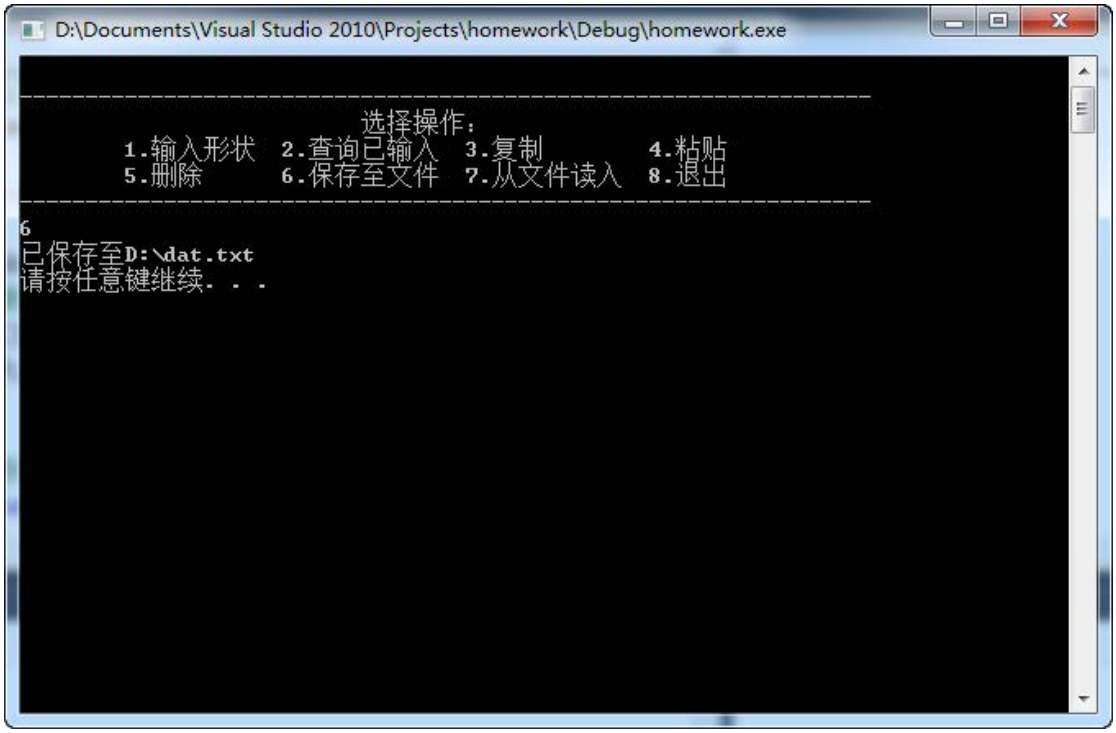
输入圆心坐标和半径，按回车键确认：



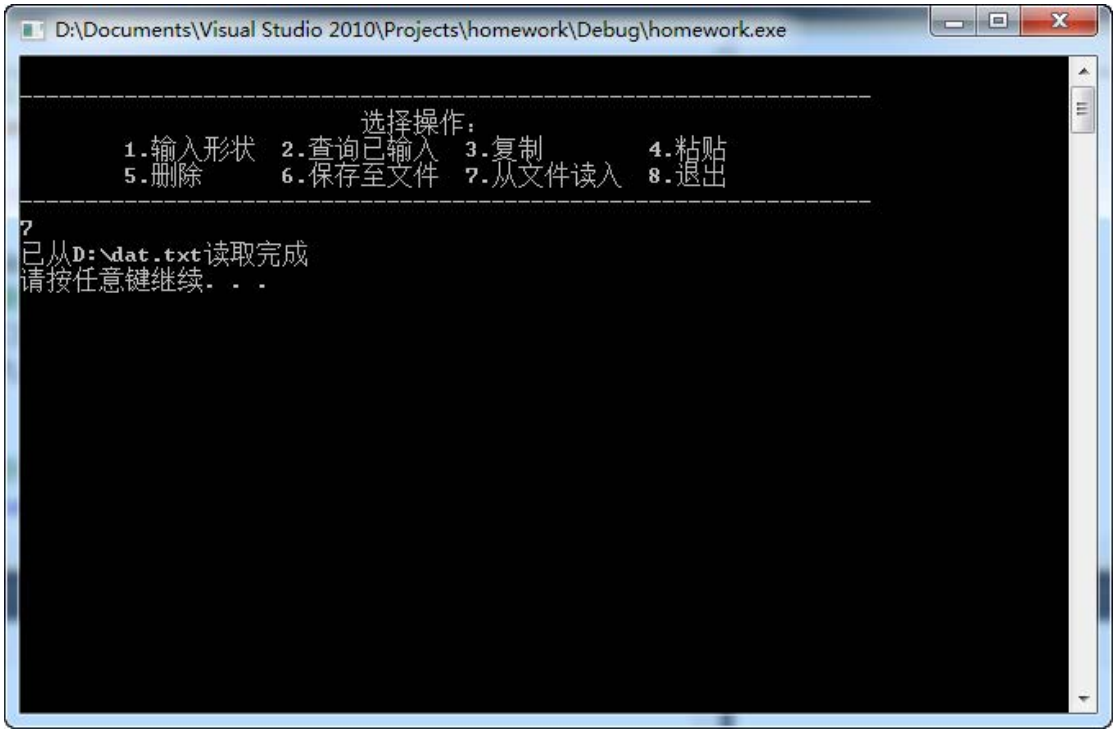
选择操作“2.查询已输入”，选择“3.显示全部”：



保存至文件（默认保存至 D:\dat.txt 中）：



从文件读取（默认从 D:\dat.txt 中读取）：



四、部分源码

```
// 多边形的存在性判断
int CPolygon::exist()
{
    // 此函数没有考虑输入的点重合时的情况
    float *length = new float[edges];
    float lengthAll = 0;
    // 计算各边长
    for(int i = 0; i < edges; i++) {
        if(i == 0) {
            length[i] = sqrt(pow((px[edges - 1] - px[0]), 2) + pow((py[edges - 1] -
py[0]), 2));
        } else {
            length[i] = sqrt(pow((px[i - 1] - px[i]), 2) + pow((py[i - 1] - py[i]), 2));
        }
        lengthAll += length[i];
    }
    float lengthHalf = lengthAll / 2.0;
    // 如果一边之长大于等于其他所有边之和，即该边长大于等于全长的一半，
多边形不存在
    for(int i = 0; i < edges; i++) {
        if(length[i] >= lengthHalf)
            return 0;
    }
    return 1;
}

// 输入数字进行选择
int input(char *promote, int choices)
{
    while(1) {
        cout << "\n" << promote << endl;
        int x;
        cin >> x;
        if(x > 0 && x <= choices) {
            return x;
        } else {
            clr();
            // 重置 cin 状态为 ios_base::goodbit,
            // 清空输入缓冲区中的错误数据,
            // 避免输入字母造成的死循环
            cin.clear();
```

```
            cin.sync();
            cout << "输入错误，请重新输入！" << endl;
        }
    }
}

// 清屏
void clr()
{
    system("CLS");
}

// 暂停
void pause()
{
    system("PAUSE");
}

// 选择操作
void inputOp()
{
    int r;
    while(1) {
        clr();
        r = input(
"----------------------------------------------------------------\n\
                                选择操作：\n\
          1.输入形状   2.查询已输入   3.复制          4.粘贴\n\
          5.删除       6.保存至文件   7.从文件读入   8.退出\n\
----------------------------------------------------------------", 8);
        switch(r) {
        case 1: // 输入形状
            inputShape();
            break;
        case 2: // 查询已输入
            if(!arr.getSize()) {
                cout << "没有数据！" << endl;
            } else {
                inputQuery();
            }
            break;
        case 3:  // 复制
            inputCopy();
            break;
```

```
            case 4:  // 粘贴
                inputPaste();
                break;
            case 5:  // 删除
                inputDel();
                break;
            case 6:  // 保存至文件
                saveToFile();
                break;
            case 7:  // 从文件读入
                loadFromFile();
                break;
            case 8:  // 退出
                exit(0);
            }
        }
}

// 保存至文件
void saveToFile()
{
    ofstream fs(fname);
    int num = arr.getSize();
    fs << num << endl;
    for(int i = 0; i != num; i++) {
        arr.get(i)->saveToFile(fs);
    }
    fs.close();
    cout << "已保存至" << fname << endl;
    pause();
}

// 从文件加载
void loadFromFile()
{
    arr.clear();
    ifstream fs("D:\\dat.txt", ios::in);
    if(!fs.good()) {
        // 发生错误，退出
        cout << "Error!" << endl;
        pause();
        return;
    }
    CShape *s;
```

```
    int shape;
    int num = 0;
    fs >> num;
    for(int i = 0; i < num; i++) {
        if(fs.eof()) {
            // 发生错误，退出
            cout << "Error!" << endl;
            pause();
            return;
        }
        shape = -1;
        fs >> shape;
        switch(shape) {
        case CShape::SHAPE_CIRCLE:
            s = new CCircle();
            break;
        case CShape::SHAPE_LINE:
            s = new CLine();
            break;
        case CShape::SHAPE_POINT:
            s = new CCPoint();
            break;
        case CShape::SHAPE_POLYGON:
            s = new CPolygon();
            break;
        case CShape::SHAPE_RECTANGLE:
            s = new CRectangle();
            break;
        case CShape::SHAPE_TRIANGLE:
            s = new CTriangle();
            break;
        default:
            // 发生错误，退出
            cout << "Error!" << endl;
            pause();
            return;
        }
        s->loadFromFile(fs);
        arr.add(s);
    }
    fs.close();
    cout << "已从" << fname << "读取完成" << endl;
    pause();
}
```