**参数列表**

|  |  |  |  |
| --- | --- | --- | --- |
| 变量 | 类型 | 解释 | 其他注释 |
| area | double | 场地面积 | 传参、用户输入 |
| budget | double | 总预算金额 | 传参、用户输入 |
| use\_budget | double | 终预算 | 引用值 |
| side | double | 边长 | 引用值，各形状最终参数 |
| with\_grass | bool | 是否要草坪 | 1则草地填满剩余区域，0反之 |
|  |  |  | 以上为calculate函数所需参数 |
|  |  |  |  |
| square\_side | double | 外边长 | 最大可利用方形边长 |
| count | double | 计数器 | 千分之一外边长，扩展最小单位 |
| tree\_budget | double | 树所需预算 | 必须满足，无法减少budget |
| result | string | 最终方案 | 为文字描述 |
|  |  |  | 以上为calculate函数内部定义参数 |
|  |  |  |  |
| area | 同上 |  |  |
| budget | 同上 |  |  |
| square\_side | 同上 |  |  |
| count | 同上 |  |  |
| tree\_budget | 同上 |  |  |
| use\_budget | 同上 |  | 依然作为引用 |
| side | 同上 |  | 依然作为引用 |
|  |  |  | 以上为search函数所需参数 |
|  |  |  |  |
| square\_shop | Rectangle | 方形超市 |  |
| rectangle\_locker | Rectangle | 矩形更衣室 |  |
| circular\_poor | Circular | 圆形泳池 |  |
| trapezoid\_rest | Trapezoid | 梯形休息区 |  |
| result | string | 最终方案 | 为文字描述 |
|  |  |  |  |
| shop\_area | double | 超市面积 | 用于计算造价 |
| locker\_area | double | 更衣室面积 | 用于计算造价 |
| poor\_area | double | 泳池面积 | 用于计算造价 |
| fence\_length | double | 篱笆长度 | 用于计算造价 |
| reat\_area | double | 休息区面积 | 用于计算造价 |
|  |  |  |  |
| shop\_budget | double | 超市预算 |  |
| locker\_budget | double | 更衣室预算 |  |
| poor\_budget | double | 泳池预算 |  |
| fence\_budget | double | 篱笆预算 |  |
| reat\_budget | double | 休息区预算 |  |
|  |  |  | 以上为search函数自定义参数 |
|  |  |  |  |
|  |  |  |  |
| grass\_price |  | 草造价 | All |
| tree\_price |  | 树造价 | 元/平方米 |
| shop\_price |  | 超市造价 | 元/平方米 |
| locker\_price |  | 更衣室造价 | 元/平方米 |
| poor\_price |  | 泳池造价 | 元/平方米 |
| fence\_price |  | 篱笆造价 | 元/米 |
| reat\_price |  | 休息区造价 | 元/平方米 |

测试函数：

**类**

#define PI 3.14

class Shape {

public:

virtual double area() const = 0;

};

//顶层抽象类

class Rectangle : public Shape {

public:

virtual double area() const;

void set\_attribute(double l, double w);

double get\_perimeter();

double get\_length();

double get\_width();

protected:

double length;

double width;

};

//矩形类

class Circular : public Shape {

public:

virtual double area() const;

void set\_radius(double r);

double get\_perimeter();

double get\_radius();

protected:

double radius;

};

//圆形类