类和对象

- C++和C语言的区别是，在C语言的基础上加入了面向对象编程，C++支持面向对象的编程设计
- 类是C++的核心特性，通常被称为是用户定义的类型

1什么是类和对象

```
class Box
{
    public:
        double length;
        double width;
        double getArea(); //计算矩形面积
};
Box box1;
Box box2[10];
```

- 类是抽象的数据类型，包含了数据的表示以及用于处理数据的方法
- 对象是类的具体***实例***
- 类中的数据和方法（函数）统称为类的***成员***
- 数据定义了类的对象包括了什么
- 方法定义了在这个对象上可以执行什么样的操作
- 类成员的访问属性
  - public 说明成员在外部是可访问的，访问方式是使用"."操作符
  - private 说明成员在外部是可访问的
  - protected public 成员在类内和派生类（子类）中可访问。
- 声明类的对象和声明基本类型变量一样
- 声明一个类的对象 Box box1;

2类的成员变量

类的对象的公共数据成员可以直接使用成员访问运算符“.”来解决

3类的成员函数

- 类的成员函数是把定义写在类定义内部的函数，就像类定义中的其他变量一样
- 类成员函数是类的一个成员，类的任意对象都可以使用类成员函数，类成员函数可以访问对象中的所有成员
- 下图分别是在类的内部定义函数以及类的外部定义函数，外部定义函数需要用到范围解析运算符“::”

```
class Box
{
    public:
        double length;
        double width;
        double getArea(); //计算矩形面积
        double getParimeter()  // 在类的内部定义函数
        {
            return 2*(length + width);
        }
};
Box box1;
Box box2[10];
double Box::getArea() //在类的外部定义函数
{
    return length*width;
}
```

- 调用成员函数，在对象后使用“.”运算符即可调用该对象的成员函数

4类的私有成员

- 声明为public的成员叫做共有成员，声明为private的成员叫做私有成员
- 共有成员在程序外中类的外部是可以访问的，可以使用成员函数来设置和获取非公有变量的值
- 私有成员的变量或者函数在类的外部是不可直接访问的，***默认情况下***，类的成员属性为 ***私有成员***,结构体默认成员类型为 ***共有成员***。
- 在实际操作中，一般会在私有区域定义数据，在共有区域定义相关函数，以便在类的外部也可以调用这些函数
- 在C++中可以使用this指针来访问自己

```cpp
#include <iostream>
#include <string>

class Person {
private:
    std::string name;
    int age;

public:
    // 设置姓名
    void setName(const std::string& newName) {
        name = newName;
    }

    // 获取姓名
    std::string getName() const {
        return name;
    }

    // 设置年龄
    void setAge(int newAge) {
        age = newAge;
    }

    // 获取年龄
    int getAge() const {
        return age;
    }
};

int main() {
    // 创建一个 Person 对象
    Person person;

    // 使用成员函数设置私有属性
    person.setName("Alice");
    person.setAge(30);

    // 使用成员函数获取私有属性并显示
    std::cout << "Name: " << person.getName() << std::endl;
    std::cout << "Age: " << person.getAge() << std::endl;

    return 0;
}
```

4类的构造函数与析构函数

```cpp
#include <iostream>

class Box {
private:
    double length;
    double width;
    double height;

public:
    // 构造函数
    Box(double l, double w, double h)
        : length(l), width(w), height(h) {
        std::cout << "Box created with dimensions: " << length << "x" << width <<
"x" << height << std::endl;
    }

    // 析构函数
    ~Box() {
        std::cout << "Box destroyed. Goodbye!" << std::endl;
    }

    // 获取体积
    double getVolume() const {
        return length * width * height;
    }
};

int main() {
    // 创建一个 Box 对象
    Box myBox(3.0, 4.0, 5.0);

    // 计算并显示体积
    std::cout << "Volume of the box: " << myBox.getVolume() << std::endl;

    return 0;
}
```

4.1类的构造函数

- 目的：构造函数用于初始化对象的数据成员，在对象创建时自动调用，确保对象在被使用前具有合适的初始状态。
- 名称：构造函数的名称与类名称相同,且没有返回值，也不会返回void。
- 特点：可以有多个构造函数，它们可以根据 ***参数的类型和数量进行重载***，以适应不同的对象初始化需求。
- 行为：在对象创建时被调用，完成对象的初始化工作。没有返回类型，并且不能手动调用，只能在对象创建时由系统自动调用。

4.2类的析构函数

- 目的：析构函数用于在对象被销毁时释放资源、进行清理操作，在对象生命周期结束时自动调用。
- 名称：构析函数的名称与类名称相同，前面加上波浪号 ~。
- 特点：每个类只能有一个析构函数，没有参数，没有返回值，不能被继承或重载。
- 行为：在对象销毁时（例如作用域结束、delete 操作等）由系统自动调用，用于跳出程序（比如关闭文件，释放内存）前释放对象占用的资源，执行清理操作。