1 立方体案例
2 点和圆关系案例
2.1 圆内的属性里有个其它的自定义数据类型 Point
2.2 三种关系判断
2.3 分文件编写
2.3.1 .h 中写类的成员函数声明
2.3.2 .cpp 中写成员函数实现
3 对象的初始化和清理
3.1 构造函数
3.1.1 没有返回值 没有 void，类名相同，可以发生重载，可以有参数
3.2 析构函数
3.2.1 没有返回，没有 void ，函数名称： ~类名，不可以发生重载，不可以有参数
3.3 系统会默认调用 构造函数和析构函数，而且只会调用一次
3.4 如果程序员没有提供构造和析构，系统会默认提供，空实现
4 构造函数的分类及调用
4.1 按照参数分类
4.1.1 无参构造（默认构造） 有参构造
4.2 按照类型分类
4.2.1 普通构造函数 拷贝构造函数
4.3 无参构造写法 和调用
4.3.1 Person p1 ; 注意不能写 Person p1() ，因为编译器认为这个是函数声明
4.4 有参构造写法 和调用
4.4.1 Person p2(10) 或者 Person p2 = Person(10)
4.4.2 Person(10) 匿名对象 ，执行当前行后就会释放这个对象
4.5 拷贝构造函数
4.5.1 Person(const Person & p)
4.5.2 Perons p1(p2) 或者 Person p1 = Person(p2)
4.5.3 不能用拷贝构造函数初始化匿名对象
4.5.3.1 如果写成 Person (p1) 这种写法等价于 Person p1
4.5.3.2 写到右值可以做拷贝构造函数
4.6 Person P = 100 隐式类型转换 相当于调用 Person p = Person(100)
5 拷贝构造函数调用时机
5.1 1、用已经创建好的对象来初始化新的对象
5.2 2、以值传递的方式给函数参数传值
5.3 3、以值方式返回局部对象
5.4 release 默认下会做优化
6 构造函数的调用规则
6.1 如果提供了有参的构造，那么系统就不会提供默认的构造了，但是会提供拷贝构造
6.2 如果提供了拷贝构造函数，那么系统就不会提供其他的构造函数了
7 深拷贝与浅拷贝
7.1 系统默认提供的拷贝构造 会进行简单的值拷贝
7.2 如果属性里有指向堆区空间的数据，那么简单的浅拷贝会导致重复释放内存的异

常

7.3 解决上述问题，需要我们自己提供拷贝构造函数，进行深拷贝

8 初始化列表语法

8.1 在构造函数后面 + ：属性(值、参数), 属性（值、参数）…

9 类对象作为成员的案例

9.1 当类对象作为类的成员时候，构造顺序是先构造类对象的构造，然后构造自己，

9.2 析构顺序与构造相反

10 explicit 关键字

10.1 作用：防止构造函数中的隐式类型转换

11 new 运算符 和 delete 运算符

11.1 Person * p = new Person 会返回一个 Person 指针

11.2 默认调用构造函数，开辟空间，返回不是 void* ，不需要强制转换

11.3 delete 释放

11.4 new 对象 用 void* 去接受，释放不了对象

11.5 new 出来的是数组 ，如何释放？ delete [] …

11.6 new 出来的是数组，肯定会调用默认构造