

# 问题

---

本节主要整理一下关于构造函数和析构函数的几个问题。

## 初识析构函数

---

- 析构函数通过在函数名前面加一个**位取反符号** ~，如 ~Cat()
- 析构函数名与类名相同，一个类只能有一个析构函数，不能重载
- 析构函数不能有任何参数，也没有返回值
- 若没有写析构函数，编译器会自动生成一个默认的析构函数
- 当对象的生命周期结束时，系统会自动调用析构函数

## 构造函数为什么一般不定义为虚函数？

在 [14\\_虚函数详解](#) 那一节中也有提到，**构造函数不能是虚函数**，原因是构造函数在创建对象时必须确定**对象类型**。具体的理解就是：

- ①首先，在**创建一个对象时必须确定其类型**。因为类型规定了对象可以进行哪些操作，所以创建对象时必须确定类型，以防止一些不恰当的操作，否则编译器就会报错。
- ②其次，因为虚函数是在运行时才确定对象的类型的，**如果构造函数声明为虚函数，那么在构造对象时，由于这个对象还没创建成功，编译器就不知道对象的实际类型**（比如基类还是派生类），就会报错。
- ③另外，从内存的角度来看，虚函数的调用需要虚表指针，而该指针存放在内存空间中，**如果构造函数声明为虚函数，由于对象还没创建，就没有内存空间，因此就没有虚表指针来调用虚函数（构造函数）了**。

## 析构函数为什么一般是虚函数？

为什么这里说的是一般，而不是必须？因为析构函数可以不是虚函数，也可以是虚函数。

- ①C++默认的析构函数不是虚函数。因为虚函数需要虚函数表和虚表指针，会占用额外的内存空间。**对于没有派生类的基类而言，将析构函数定义为虚函数就会浪费内存空间**。
- ②如果存在派生类继承了基类，而基类的析构函数不是声明为虚函数，那么**在析构一个指向派生类的基类指针时，就只会调用基类的析构函数，不会调用派生类的析构函数，因此会造成内存泄漏的问题**。

## 子类在析构时，要调用父类的析构函数吗？

**不需要自己显示地调用基类的析构函数**，因为编译器会自动调用，如果再显示地调用，就会调用了两次，可能会出现意外的错误。

析构函数在析构时的顺序是，**先析构派生类然后析构基类**，也就是说在析构基类的时候，派生类的全部信息已经销毁了。析构函数调用的顺序与构造函数的顺序刚好相反。

