1、 内存管理的基本概念及范围

问题？

1、为什么要有内存管理？

2、内存管理主要是对内存中的哪个区进行管理？为什么？

3、内存管理的范围

什么是内存管理:对象如果不再使用,就应该回收她的空间,防止内存泄露

1.为什么要有内存管理

全局变量多,会占据BSS段和数据区,导致程序启动会加载很多全局变量,导致程序启动很慢

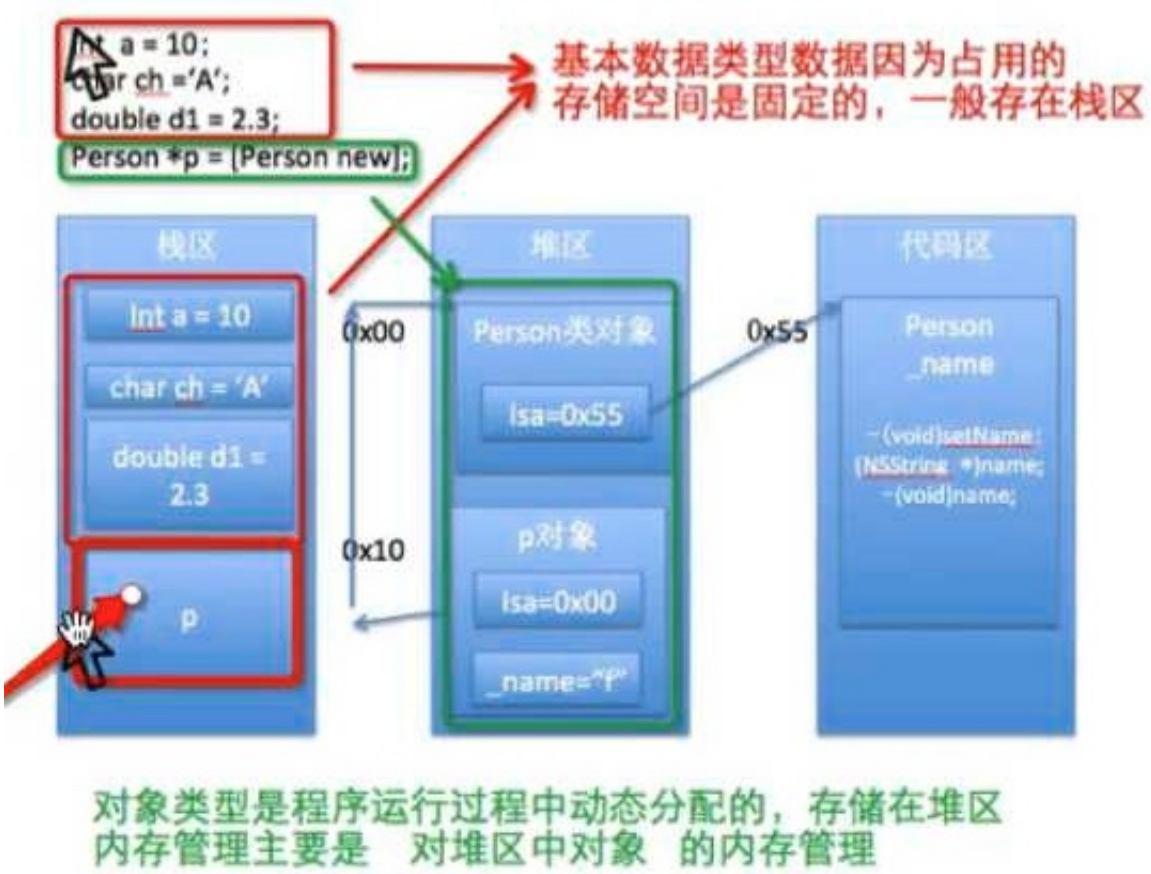
2.哪个区管理?为什么?

3.管理范围:

管理范围：

管理任何继承NSObject的对象，对其他的基本数据类型无效。

本质原因是因为对象和其他数据类型在系统中的存储空间不一样，其它局部变量主要存放于栈中，而对象存储于堆中，当代码块结束时这个代码块中涉及的所有局部变量会被回收，指向对象的指针也被回收，此时对象已经没有指针指向，但依然存在于内存中，造成内存泄露。



间接继承算不算?也算是被管理的对象

2、内存管理的原理及分类

- 问题:
  1，什么是引用计数器?
  2，系统是通过什么来管理一个对象的内存的?
  3，系统如何对引用计数器的操作

1、OC内存管理的原理

1）对象的所有权及引用计数

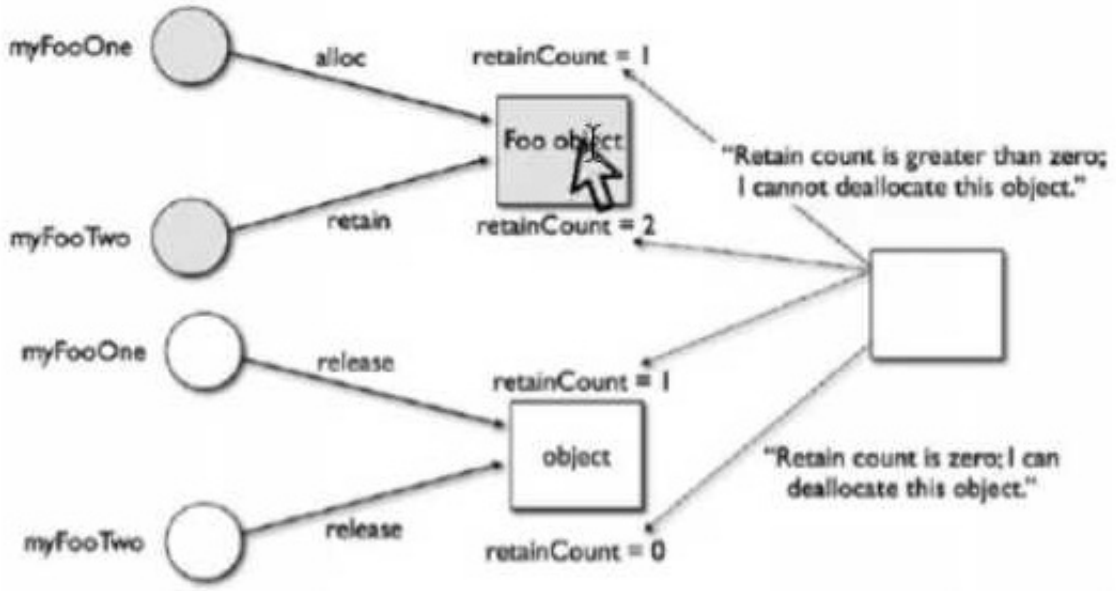
对象所有权概念

任何对象都可能拥有一个或多个所有者。只要一个对象至少还拥有一个所有者，它就会继续存在

```
Person *p = [[Person alloc] init];
```

Cocoa所有权策略

任何自己创建的对象都归自己所有，可以使用名字以“alloc”或“new”开头或名字中包含“copy”的方法创建对象，可以使用retain来获得一个对象的所有权



retainCount

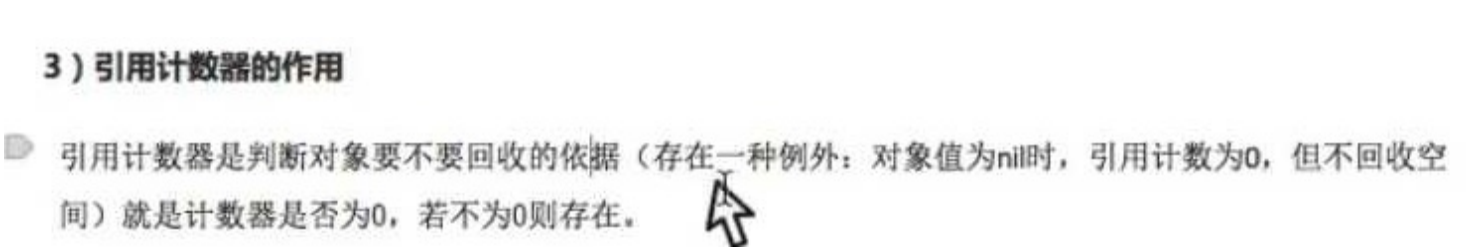
1.什么是引用计数器:有多少所有者在使用对象

对象的引用计数器

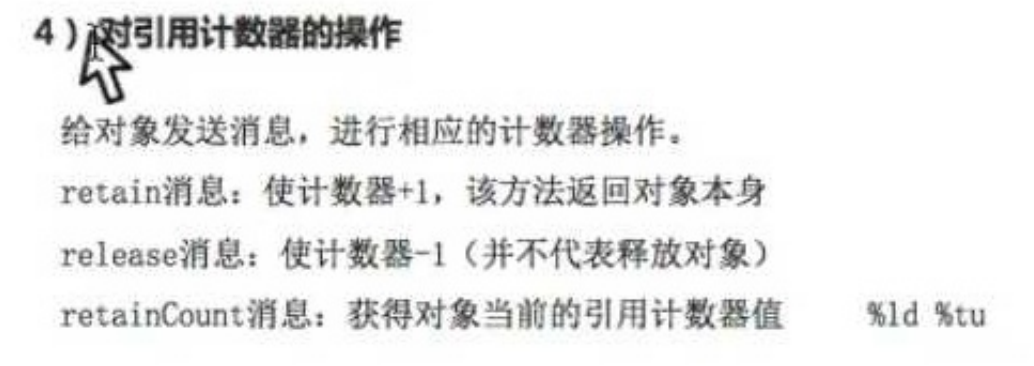
每个OC对象都有自己的引用计数器，是一个整数表示对象被引用的次数，即现在有多少东西在使用这个对象。对象刚被创建时，默认计数器值为1，当计数器的值变为0时，则对象销毁。

在每个OC对象内部，都专门有8个字节的存储空间来存储引用计数器。

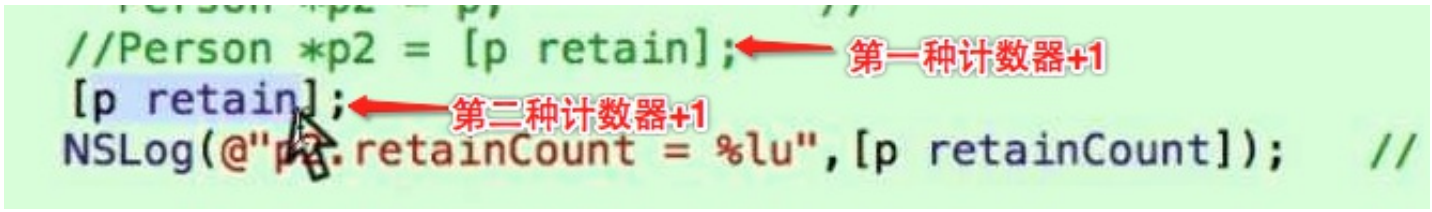
2.系统通过什么来管理内存?:引用计数器



3.系统如何对引用计数器进行操作?:通过retain和release,对计数器进行+1和-1的操作



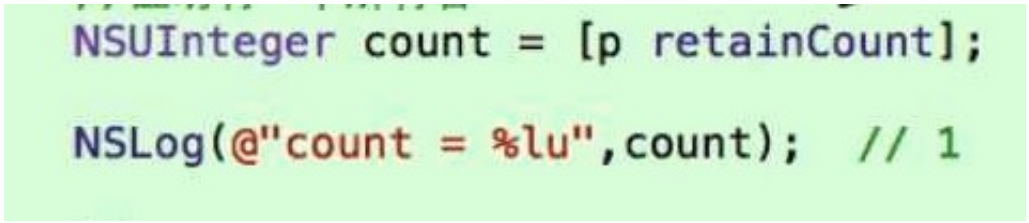
如何使计数器+1;



如何使计数器-1;计数器有多少就减几次,

retainCount是long类型,所以占用8个字节

如何查看计数器的值



5) 对象的销毁

当一个对象的引用计数器为0时，那么它将被销毁，其占用的内存被系统回收。

当对象被销毁时，系统会自动向对象发送一条dealloc消息，一般会重写dealloc方法，在这里释放相关的资源，dealloc就像是对象的“临终遗言”。

一旦重写了dealloc方法就必须调用[super dealloc]，并且放在代码块的最后调用（不能直接调用dealloc方法）。

一旦对象被回收了，那么他所占据的存储空间就不再可用，坚持使用会导致程序崩溃（野指针错误）。

注意：

1）如果对象的计数器不为0，那么在整个程序运行过程，它占用的内存就不可能被回收(除非整个程序已经退出)

2)任何一个对象，刚生下来的时候，引用计数器都为1。（对象一旦创建好，默认引用计数器就是3)当使用alloc、new或者copy创建一个对象时，对象的引用计数器默认就是1

如何查看对象是否被销毁

系统会自动,也可以在子类中重写一次,看是否真的销毁,且不用声名!!,直接去.m中写实现方法

```
//dealloc方法,是对象的临终遗言的方法
//对象被销毁的时候,会默认的调用该方法
- (void)dealloc
{
    //1 先释放子类自己的对象的空间
    NSLog(@"Person已经挂了");
    //2 再释放父类的
    [super dealloc];
}
@end
```

super dealloc必须有

2、OC内存管理分类

Objective-C提供了三种内存管理方式：

1 MannulReference Counting（MRC，手动管理，在开发 iOS4.1之前的版本的项目时我们要自己负责使用引用计数来管理内存，比如要手动 retain、release、autorelease 等，而在其后的版本可以使用 ARC，让系统自己管理内存。）

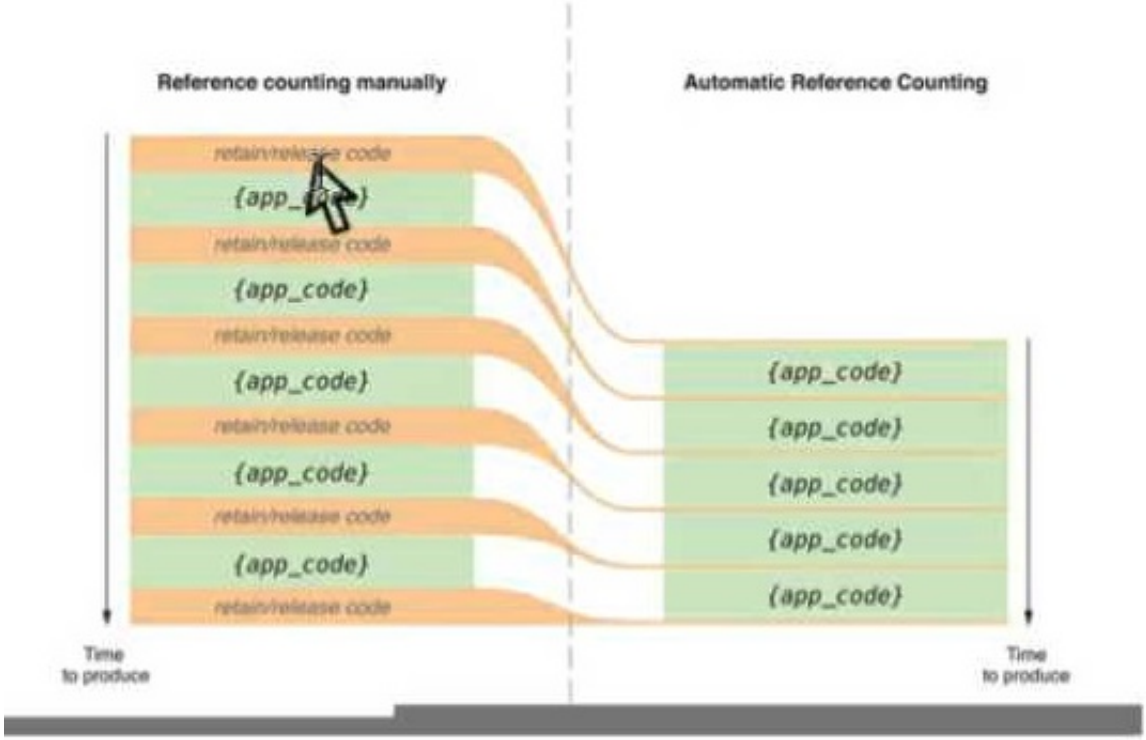
2 automatic reference counting（ARC，自动引用计数，iOS4.1 之后推出的）

3 garbage collection（垃圾回收）。iOS不支持垃圾回收；

ARC作为苹果新提供的技术，苹果推荐开发者使用ARC技术来管理内存；

开发中如何使用：需要理解MRC，但实际使用时尽量用ARC

第三种在IOS中不使用!!!



3-【掌握】手动内存管理快速入门

本小节知识点：

1、【掌握】关闭ARC的方法

2、【掌握】手动管理（MRC）快速入门

问题：

1，对象释放时，系统默认会调用哪个方法？

对象释放时系统默认自动调用NSObject中的dealoc方法,不需要手动调用,不过可以在类中进行重写

注意
1)永远不要直接通过对象调用dealloc方法(实际上调用并不会出错)
一旦对象被回收了，它占用的内存就不再可用，坚持使用会导致程序崩溃(野指针错误)为了防止调用出错，可以将“野指针”指向nil(0)。

4-【理解】内存管理的原则

本小节知识点：

1、【理解】内存管理的原则

2、【理解】内存管理研究的内容

- 问题：
  1，内存管理的原则有哪几个？
  2，内存管理研究的内容

1.内存管理的原则:

1、内存管理的原则

1）原则

只要还有人在使用某个对象，那么这个对象就不会被回收；

只要你想使用这个对象，那么就应该让这个对象的引用计数器+1；

当你不想使用这个对象时，应该让对象的引用计数器-1；

-1后谁不能用?怎么判断?

2）谁创建，谁release

（1）如果你通过alloc, new, copy来创建了一个对象，那么你就必须调用release或者autorelease方法

（2）不是你创建的就不用你去负责

3）谁retain，谁release

只要你调用了retain，无论这个对象时如何生成的，你都要调用release

4）总结

有始有终，有加就应该有减。曾经让某个对象计数器加1，就应该让其在最后-1.

2.内存管理研究的内容:

内存管理研究的内容：

1)野指针： 1)定义的指针变量没有初始化 2)指向的空间已经被释放了

2)内存泄露：

```
Person *p  = [Person new];
p 栈区
[Person new];  堆区
```

如果栈区的p已经释放了，而堆区的空间还没有释放，堆区的空间就被泄露了