1.定义一个类

@interface 类名 : NSObject

{

属性;

}

@end

2.实现一个类

@implementation 类名

@end

3.创建一个对象

类名 \*对象名 = [类名 new];

4.访问对象的属性(默认情况下，对象的属性是不允许被外界调用的，需要在属性上添加 @public 关键字)

属性赋值 ：对象名 -> 属性名 = 值

属性取值：对象名 -> 属性

5.无参方法

方法的声明

位置：在@interface的大括号外面，@end的里面

-(返回值类型)方法名;

-(void)study;没有返回值，无参的方法，名字叫做study

方法的实现：

位置：在@implementation 之中实现

把声明中的方法拷贝过来,把分号改成大括号{},其中放方法实现

方法的调用：

方法无法直接被调用，必须先创建一个对象，由对象来调用方法

[对象名 方法名];

6.带1个参数的方法

-(返回值类型)方法名:(参数类型)参数名;

调用：

[对象名 方法名:参数内容];

7.带多个参数的方法

-(返回值类型)方法名：(参数类型)参数1 :(参数类型)参数2

调用：

[对象名 方法名:参数1内容:参数2内容];

**五、内存中的五大区域**

1.栈:存储局部变量

2.堆:程序员手动申请的内存空间

3.BSS段:存储未被初始化的全局变量 静态变量

4.数据段:初始化完的全局、静态、常量数据

5.代码段:存储代码，存储类的代码

**六、类的加载**

1.创建对象时，必需要访问类

2.声明一个类的指针变量也会访问类

在程序运行期间，当某个类被第 1 次访问到的时候，会将这个类存储到内存中的代码段中，这个过程就称为类加载

在类的第 1 次访问时，才会做类加载

-但类被加载后，直到程序结束的时候才会被释放

Person \*p:会在栈中申请一块空间，存储Person类型的指针变量p

[Person new]:真正的在内存中创建对象

New:在堆中申请一块合适的大小的空间，再根据类的模板创建对象，模板中定义了哪些属性就依次声明在对象中。

对象中还有另外的一个隐藏属性，叫ISA指针，指向对象所属的类在代码段中的地址

//对象内携带sum方法

@interface Maths:NSObject

-(int)sum:(int)i and:(int)j;//可用and with来描述

@end

**七、常见错误**

1.文件名与类名一定要一致，类名首字母大写，方法名首字母小写，按驼峰起名法

2.不能只有方法的声明，没有实现

3.类的声明一定要放在使用类的前面

4.属性不允许的声明的时候初始化

**八、OC中的方法分类**

1.对象方法：先创建对象后，才能调用对象方法

2.类方法：类方法调用不需要依赖于对象，可以直接通过类名来调用，不能在类方法中访问属性

类方法的声明：使用+号

类方法的调用：[类名 类方法]

类方法特点：

1.节约空间：无需创建对象

2.提高效率：调用方法无需拐弯，直接找到代码段中的类方法执行

类方法的规范：

1.写一个类，苹果要求为这个类提供一个同名的类方法，这个类方法帮我们创建一个纯净的对象返回

**九、OC中的NSString**

1.NSString是一个字符串类型，用来保存OC字符串

2.NSString是Foundation框架中的一个类

NSString 3种创建

1.NSString \*str=@"abc";

2.NSString \*str2= [NSString new];

str2=@"bcd";

3.NSString \*str3 = [NSString string];

str3=@"efg";

NSString的类方法：

1.stringWithUTF8String:把C语言字符串转化为OC字符串

char \*str4 = "xyz";

NSString \*str5 = [NSString stringWithUTF8String:str4];

2.stringWithFormat:

拼接字符串

NSString的对象方法

1.length:字符串长度

str= @ "a";

NSUInteger len = [str length]

2.characterAtIndex 通过下标取字符

str = @"abcd";

unichar \*ca = [str characterAtIndex:2];

NSLog(@"第2位是:%C",ca);

3.isEqualToStrign:比较字符串内容是否相等

[str1 isEqualToString:str2]

Equal比较的是内容

双= 比较的是地址（存在堆中的） (基本数据类型如int时，存在栈中，双=比较的也是值)

4.compare:比较字符串的大小

（相等0，比它大1，比它小-1）

str1 = @"d";

str2 = @"b";

int res = [str1 compare:str2];

**十、OC中的匿名对象**

没有名字的对象，创建一个对象，没有用指针存放它，这个对象就叫匿名对象

[Person new];

[Person new]->\_name = @"jack";

[[Person new] eat];

匿名内部类只能使用1次，每次创建都是新的对象

**十一、封装**

1.类其实就是一个高级的封装，把数据和行为封装成一个整体

2.屏蔽内部的实现，外界不需要知道内部如何实现，只需要知道如何使用。

3.后期维护特别方便

**十二、OC中封装具体实现**

1.先把属性私有化，不给外部（其他类）随意修改。去掉属性上的@public

2.为属性添加setter方法，修改属性必须调用其set方法。

a)该方法为对象方法

b)该方法没有返回值，但一定要有传入参数

c)方法名必须以set开头，去掉\_，首字母大写

d)在方法中，添加实现逻辑，保证数据的合理性。

3.为属性添加取值方法

a)该方法为对象方法

b)该方法必须有返回值（返回值类型与属性类型一致），不用传入参数

c)方法名直接为属性名（去掉\_）

**十三、OC中static关键字**

1.static不能修饰属性和方法

2.只能修饰方法中的局部变量

3.修饰后的局部变量会变成静态变量，存储在常量区，当方法执行完不会回收

**十四、OC中self关键字**

1.相当于Java中的this关键字

2.可以在对象方法和类方法中使用

3.在对象方法中self指向当前对象，谁调用这个方法谁就是当前对象

4.想调用同名的属性和方法时要使用self关键字

5.在类方法中self指向当前类

**十五、OC中继承**

1.子类继承父类，子类会拥有父类所有的属性和方法，子类有自己独有的属性和方法

2.必须满足条件 is a

3.OC是单继承

4.存在传递性，A类从B类继承，B类从C类继承，A类就拥有B C所有的属性和方法

**十六、OC中super关键字**

1.可以在对象方法和类方法中使用

2.在对象方法中，super可以调用从父类继承的方法

**十七、OC中的多态**

1.多态的关键是重写

2.父类引用指向子类对象：Animal \*an = [Dog new];

**十八、OC中重写**

1.必须先有继承

2.子类无需声明父类的方法，直接在子类实现中重写父类的方法

3.方法名，参数，返回值必须一样

**十九、OC中访问修饰符（考试可能考）**

1.@private ：私有，被private修饰的**属性**只能在本类中使用

2.@protected：受保护的，被protected修饰的属性只能在本类和其子类中使用

3.@package：被package修饰的只能在当前框架中使用

4.@public ：公共的，被public修饰的属性可以在任意地方使用

5.OC中不加访问符默认是protected

**二十、OC中类对象Class**

1.存储在代码段当中的类对象

Class cla = [Person class];

Person \*p1 = [cla new];

**二十一、OC中真私有**

1.属性不放在声明@interface中 ，而把它放在实现@implementation中

2.外面无法在再查看类中的属性

3.方法私有化：把方法放在实现中，只能给本类中的其他方法调用。

**二十二、OC中@property**

1.帮我们自动生成getter 和setter方法的**声明，**写在@interface中

语法：@property 数据类型 名称,…,… ;（此时不用加下划线，可以批量使用）

@property NSString \*name;

**二十三、OC中@synthesize（通常与property成对）**

1. 帮我们自动生成getter 和setter方法的**实现，**写在@implementation中

语法：@synthesize 名称=\_名称;（不会自动帮你补下划线，也可批量）

**二十四、OC中（XCode4.4后）增强@property**

1. XCode4.4后增强@property

2.只要写了@property，自动生成属性，自动生成get set方法 的 声明加实现

**二十五、OC中new**

1. 先alloc 帮我们创建一个对象，再init 初始化这个对象，返回对象的地址

Person \*p = [Person new];  
相当于

Person \*p1 =[ [Person alloc]init];

1. init 方法又称构造方法

返回类型为instancetype

（必考，试卷倒数1、2题）

-(instancetype)init{

self = [super init];

if(self!=nil){ //或if(self = [super init]{…})

self->name=@”abc”;

self.age=9;

}

return self;

}

直接打印：在实现重写description

-(NSString\*)description{

Return [NSString stringWithFormat:@”name:%@”,\_name];

}

主函数中：

NSLog(@”p:%@”,p);

**二十六、OC中内存管理（考试、面试）**

1.引用计数器retainCount：当new一个对象，引用计数器为1，当多一个人调用该对象，引用计数器+1，当少一个人使用，引用计数器-1，当为0时，系统自动回收该对象。

2.MRC：手动内存管理，手动管理引用计数器

3.ARC：自动内存管理（默认），自动管理引用计数器

4.retain：引用计数器+1

5.release：引用计数器-1，当为0时，会调用对象的dealloc方法

-(void)dealloc{//销毁对象的方法，重写在实现中

NSLog(@”%@ is dead. ”,\_name);

[super dealloc];

}

**二十七、OC中野指针和僵尸对象**

1.回收对象：把对象占用的空间分配给别人

当对象占用的空间还没有分配给别人时，其实对象数据还存在。

2.OC中野指针：指向的对象已经被回收了，这个指针称为野指针

3.僵尸对象：一个已经被销毁的对象，但是这个对象占用的空间还没有被回收分配给别人（野指针指向的对象称为僵尸对象）

4.OC中有僵尸对象检查机制，打开后（高性能耗），只要访问僵尸对象就会报错。