简答题:

1.请写出ARC 的中英文全称

答: automatic reference counting

2.内存管理的原则是什么?

答:凡是调用了alloc、new、copy、retain等语法创建对象，都对应地调用一次release或autorelease为引用计数器减一。

3.什么时候对象内存计数器会增加？

答:①创建对象的时候；②调用retain方法的时候。

4.创建一个对象需要经过那三个步骤？

答:1.分配内存；2.调用父类init方法；3.初始化成员变量。

5. .什么是野指针?什么是僵尸对象?什么是空指针?

答：

野指针：指向了不可访问的对象的指针变量。

僵尸对象：没有指针指向的内存对象；

空指针：值为nil的指针变量。

6. Set方法的作用是什么？Set方法的好处？

答：

作用：为成员变量赋值

好处：1.实现对成员变量的封装：实现程序员想要的业务逻辑；2.管理内存：实现内存平衡。

7. Get方法的作用是什么？

答：1.获取属性值。2.返回自动释放内存的对象(该对象属于该类的属性的时候)。

8. 结构体当中能定义oc对象吗？

答：手动内存管理中能，ARC中不能。

9.什么是构造方法，使用构造方法有什么注意点。

答:构造方法：用于创建对象初始化成员变量的方法就是构造方法，分为类构造方法和对象构造方法。注意点：要先调用父类init方法，然后赋值给self，最后返回self。

10.请说明NSSet、NSArray、NSDictionary的区别?

答:相同点：1.都是不可变的；2.都属于集合类，用于存放一个或多个对象。

不同点：

1. NSSet是无序的，NSArry和NSDictionary是有序的。
2. NSSet只能一次性读取，不能指定索引读取；NSArry和NSDictionary可以根据索引读取对应的值。
3. 被释放的时候都会对内部元素进行一次release操作。

11.请说明 #include、#import @class各自的作用

答：

#include 是C语言预处理指令，其作用仅仅是拷贝对应的文件内容

#import 是objective-C预处理指令，不仅仅能拷贝对应的文件内容，还能避免重复包含。

@class 告诉编译器其后紧跟的是一个类。如果要使用该类内部的属性或方法，要使用#import代替@class。

12.点语法本质是什么,写一个点语法的例子,并写上注释

答:

点语法的本质是方法。

/\*\*

\* 获取Name的值

\*/

-(NSString \*)name{

return \_name;

}

/\*\*

\* 打招呼

\*/

-(void)sayHello{

NSLog(@"名为%@的人给你打招呼了！",self.name);

}

13.@property 有什么作用

答:@property 是编译器语法，告诉编译器生成对应的getter、setter方法和成员变量。

14.调用对象的release 方法会销毁对象吗？

答： 不会销毁对象，会对对象引用计数器-1操作，当retainCount等于0的时候才会立即销毁对象。

15.请说明自动释放池autorelease的作用，和工作机制，它与release有什么不同？

答：

作用：延长对象内存释放时间。

工作机制：将对象存入栈顶的自动释放池，当自动释放池被销毁的时候会对池中的对象进行一次release操作。

与release的不同点：release是让计数器立即减一。

16.请简述什么是协议？

答:

协议是对象之间共同遵守的契约，协议内容就是契约条款，这些条款都是一个个的方法声明。

17.id类型是什么，instancetype是什么，有什么区别？

答：

id类型是万能指针，可以指向任何对象；

instancetype和id类似，用法上有些区别。

区别：1.instancetype只能用于方法返回值，id可以用于定义对象，方法的形参、方法的返回值。2.当id做方法返回值的时候，编译器不会主动检测真实类型，而instancetype会进行真实类型检测。

18.成员变量名的命名以下划线开头的好处？

答： 1. 便于使用，当输入下划线的时候提示会比较准确；2.在方法内部使用的时候，可以方便区分成员变量和参数。

19.Self的用途是什么？

答： self代表当前对象。可以用于实现多态特性，还可以用于访问成员变量和成员方法。

20. Super关键字的作用？

答：Super代表父类对象，可以用于访问父类成员变量或调用父类方法。

编程题:

1. 查找字符串位置

给NSString 增加个分类,获取字符串http://www.itcast.cn中的itcast开始下标与结束下标的位置（字符串：）

//头文件

#import <Foundation/Foundation.h>

@interface NSString (Extension)

/\*\*

\* 查找字符串所在开始下标和结束下标

\* @param stringToSearch 要查找的字符串

\* @return 下标 location代表开始下标 length代表结束下标

\*/

-(NSRange)startIndexAndEndIndexOfString:(NSString \*)stringToSearch;

@end

//实现文件

#import "NSString+Extension.h"

@implementation NSString (Extension)

/\*\*

\* 查找字符串所在开始下标和结束下标

\* @param stringToSearch 要查找的字符串

\* @return 下标 location代表开始下标 length代表结束下标

\*/

-(NSRange)startIndexAndEndIndexOfString:(NSString \*)stringToSearch{

NSRange range={-1,0};

NSString\* str=self;

NSRange tempRange=[str rangeOfString:stringToSearch];

if (tempRange.location!=NSNotFound) {

range.location=tempRange.location;

range.length=tempRange.location+tempRange.length-1;

}

return range;

}

@end

2. 设计一个形状类，提供（计算、显示）面积和周长的方法，设计一个长方形类和正方形类 继承形状类，并给长方形类提供长度和宽度成员属性，给正方形类提供边长属性，程序运行 输出长方形和正方形的面积和周长。

//TJQShape头文件

#import <Foundation/Foundation.h>

@interface TJQShape : NSObject

//1.0 计算并打印面积

/\*\*

\* 计算并打印面积

\* @return 面积

\*/

-(double)areaAndPrint;

//2.0 计算并打印周长

/\*\*

\* 计算并打印周长

\* @return 周长

\*/

-(double)circumferenceAndPrint;

@end

//TJQShape实现文件

#import "TJQShape.h"

@implementation TJQShape

/\*\*

\* 计算并打印面积

\* @return 面积

\*/

-(double)areaAndPrint{

NSLog(@"请重写此方法：%s",\_\_func\_\_);

return -1;

}

/\*\*

\* 计算并打印周长

\* @return 周长

\*/

-(double)circumferenceAndPrint{

NSLog(@"请重写此方法：%s",\_\_func\_\_);

return -1;

}

@end

//TJQRectangle头文件

#import <Foundation/Foundation.h>

#import "TJQShape.h"

@interface TJQRectangle:TJQShape

/\*\*

\* 长度

\*/

@property(nonatomic,assign)double length;

/\*\*

\* 宽度

\*/

@property(nonatomic,assign)double width;

/\*\*

\* 构造函数

\* @param len 长度

\* @param width 宽度

\*

\* @return 实例化对象

\*/

-(instancetype)initWithLength:(double)len andWidth:(double)width;

@end

//TJQRectangle实现文件

#import "TJQRectangle.h"

@implementation TJQRectangle

/\*\*

\* 构造函数

\* @param len 长度

\* @param width 宽度

\* @return 实例化对象

\*/

-(instancetype)initWithLength:(double)len andWidth:(double)width{

if (self=[super init]) {

\_length=len;

\_width=width;

}

return self;

}

//1.0 计算并打印面积

/\*\*

\* 计算并打印面积

\* @return 面积

\*/

-(double)areaAndPrint{

double area=-1;

//1.0 计算面积

area=self.length\*self.width;

//2.0 打印面积

NSLog(@"面积：%f",area);

// 3.0 返回面积

return area;

}

//2.0 计算并打印周长

/\*\*

\* 计算并打印周长

\* @return 周长

\*/

-(double)circumferenceAndPrint{

double circumference=-1;

//1.0 计算周长

circumference=(self.length+self.width)\*2;

//2.0 打印周长

NSLog(@"周长：%f",circumference);

// 3.0 返回周长

return circumference;

}

@end

//TJQSquare头文件

#import "TJQShape.h"

@interface TJQSquare : TJQShape

/\*\*

\* 边长

\*/

@property(nonatomic,assign)int length;

/\*\*

\* 构造函数

\* @param length 边长

\* @return 实例变量

\*/

-(instancetype)initWithLength:(double)length;

@end

//TJQSquare实现文件

#import "TJQSquare.h"

@implementation TJQSquare

/\*\*

\* 构造函数

\* @param length 边长

\* @return 实例变量

\*/

-(instancetype)initWithLength:(double)length{

if (self=[super init]) {

\_length=length;

}

return self;

}

/\*\*

\* 计算并打印面积

\* @return 面积

\*/

-(double)areaAndPrint{

double area=-1;

//1.0计算面积

area=self.length\*self.length;

//2.0 打印面积

NSLog(@"面积：%f",area);

//3.0返回面积

return area;

}

/\*\*

\* 计算并打印周长

\* @return 周长

\*/

-(double)circumferenceAndPrint{

double circumference=-1;

//1.0计算周长

circumference=self.length\*4;

//2.0 打印面积

NSLog(@"周长：%f",circumference);

//3.0返回周长

return circumference;

}

@end

//main函数测试代码

NSLog(@"测试长方形");

TJQRectangle \*rectangle =[[TJQRectangle alloc] initWithLength:2.0 andWidth:3.0];//测试长方形

[rectangle areaAndPrint];

[rectangle circumferenceAndPrint];

NSLog(@"测试正方形");

TJQSquare \*square=[[TJQSquare alloc] initWithLength:3];//测试正方形

[square areaAndPrint];

[square circumferenceAndPrint];

3. 设计一个日期工具类，要求获取当前日期是星期几

/\*\*

\* 当天的星期

\* @return 星期

\*/

+(NSString\*)IndexOfWeekInCurrentDate{

NSArray \*weekArry=@[@"星期日",@"星期一",@"星期二",@"星期三",@"星期四",@"星期五",@"星期六"];

NSDate \*date1=[NSDate date];

NSCalendar \*calendar=[NSCalendar currentCalendar];

NSDateComponents \*cmps=[[NSDateComponents alloc] init];

NSInteger unit=NSCalendarUnitYear|NSCalendarUnitMonth|NSCalendarUnitDay|NSCalendarUnitHour|NSCalendarUnitMinute|NSCalendarUnitSecond|NSCalendarUnitWeekday;

cmps=[calendar components:unit fromDate:date1];

NSInteger weekCount= cmps.weekday;

return [weekArry objectAtIndex:weekCount-1];

}