[01-学前须知 5](#_Toc14900969)

[02-Mac\_OS\_X介绍 5](#_Toc14900970)

[03-C语言\_31\_44 6](#_Toc14900971)

[01\_基础语法 6](#_Toc14900972)

[07-第一个C程序04-编译、链接、运行 6](#_Toc14900973)

[27-变量07-练习-变量值的互换 7](#_Toc14900974)

[28-scanf函数01-变量的内存分析 7](#_Toc14900975)

[29-scanf函数02-基本使用 7](#_Toc14900976)

[02\_基本运算 7](#_Toc14900977)

[12-逻辑运算01-简介 8](#_Toc14900978)

[03\_流程控制 8](#_Toc14900979)

[01\_选择结构 8](#_Toc14900980)

[01-if01-第1种结构 8](#_Toc14900981)

[07-switch01-基本使用 9](#_Toc14900982)

[02\_循环结构 9](#_Toc14900983)

[04\_函数 10](#_Toc14900984)

[14-main函数和printf函数 10](#_Toc14900985)

[05\_内存剖析 10](#_Toc14900986)

[01-进制01-基本概念 10](#_Toc14900987)

[10-类型说明符01-short和long 11](#_Toc14900988)

[11-类型说明符02-signed和unsigned 11](#_Toc14900989)

[12-位运算01-按位与 11](#_Toc14900990)

[06\_复杂数据类型 12](#_Toc14900991)

[01\_数组 12](#_Toc14900992)

[04-使用注意 12](#_Toc14900993)

[08-数组和函数 12](#_Toc14900994)

[02-字符串 13](#_Toc14900995)

[02-内存存储细节 13](#_Toc14900996)

[03-指针 13](#_Toc14900997)

[01-基本使用 13](#_Toc14900998)

[09-指针与数组 13](#_Toc14900999)

[10-指针与字符串 14](#_Toc14901000)

[13-返回指针的函数 14](#_Toc14901001)

[14-指向函数的指针 14](#_Toc14901002)

[04-其他数据类型 14](#_Toc14901003)

[02-结构体01-基本使用 14](#_Toc14901004)

[06-结构体05-结构体数组 15](#_Toc14901005)

[07-结构体06-指向结构体的指针 15](#_Toc14901006)

[07\_其它 16](#_Toc14901007)

[01-宏定义01-不带参数的宏 16](#_Toc14901008)

[04-Objective-C 16](#_Toc14901009)

[01\_基本语法 16](#_Toc14901010)

[02\_面向对象 17](#_Toc14901011)

[01\_思想 17](#_Toc14901012)

[02-类和对象 17](#_Toc14901013)

[02\_类和对象 17](#_Toc14901014)

[01-类的声明和实现 17](#_Toc14901015)

[03\_三大特性 19](#_Toc14901016)

[01-封装01-set方法 19](#_Toc14901017)

[14-继承01-基本使用 20](#_Toc14901018)

[19-多态01-基本使用 20](#_Toc14901019)

[04\_开发技巧 20](#_Toc14901020)

[05\_核心语法 21](#_Toc14901021)

[03-成员变量的作用域01-4种作用域 21](#_Toc14901022)

[05-@property和@synthesize01-@property 21](#_Toc14901023)

[11-构造方法02-重写init方法01 21](#_Toc14901024)

[19-分类02-使用注意 22](#_Toc14901025)

[25-类的深入研究03-类的加载和初始化 22](#_Toc14901026)

[03-内存管理 23](#_Toc14901027)

[01-为什么要管理内存 23](#_Toc14901028)

[10-set方法内存管理01-基本实现 24](#_Toc14901029)

[13-set方法内存管理04-@property的内存管理 24](#_Toc14901030)

[16-循环retain和@class 24](#_Toc14901031)

[17-autorelease01-基本使用 24](#_Toc14901032)

[25-ARC01-基本原理 25](#_Toc14901033)

[27-ARC03-Xcode的ARC转换功能 25](#_Toc14901034)

[28-ARC04-循环引用 25](#_Toc14901035)

[04\_其它 25](#_Toc14901036)

[01-block01-基本使用 25](#_Toc14901037)

[05-protocol01-基本使用 25](#_Toc14901038)

[11-protocol07-代理设计模式 26](#_Toc14901039)

[05-Foundation 26](#_Toc14901040)

[01\_牛刀小试 27](#_Toc14901041)

[02-基础控件 27](#_Toc14901042)

[01-UIView的常见属性 27](#_Toc14901043)

[02-按钮操作 27](#_Toc14901044)

[03-图片浏览器 28](#_Toc14901045)

[04-图片排列 29](#_Toc14901046)

[05-汤姆猫 29](#_Toc14901047)

[06-联系人管理 29](#_Toc14901048)

[03-高级控件 30](#_Toc14901049)

[01-UIScrollView 30](#_Toc14901050)

[02-UITableView 31](#_Toc14901051)

[01-多组数据展示 31](#_Toc14901052)

[02-单组数据展示 31](#_Toc14901053)

[03-性能优化 31](#_Toc14901054)

[04-编辑模式 32](#_Toc14901055)

基础入门

# 01-学前须知

01-什么是iOS和iOS开发

02-为什么选择移动开发

03-iOS和android对比

04-iOS开发的准备

05-iOS开发初体验

06-iOS设备发展史

# 02-Mac\_OS\_X介绍

01-Dock

02-菜单栏

03-Finder

04-常见软件

05-快捷键

06-窗口操作

07-文件预览

08-软件操作

09-文件系统

10-Unix指令

# 03-C语言\_31\_44

## 01\_基础语法

01-计算机和软件开发常识

02-计算机语言发展史

03-C语言简介

04-第一个C程序01-新建源文件

05-第一个C程序02-C程序组成

06-第一个C程序03-代码编写

### 07-第一个C程序04-编译、链接、运行

编译: cc –c 文件名.c

链接: cc 文件名.o

运行: ./a.out

编译链接: cc 文件名.c

08-第二个C程序

09-初学者常见错误

10-Unix指令补充

11-知识总结、学习方法和建议

12-多种语言对比

13-关键字

14-标识符

15-注释01-基本使用

16-注释02-其他作用

17-注释03-使用注意

18-注释04-练习和总结

19-数据

20-常量

21-变量01-定义

22-变量02-赋值

23-变量03-练习-变量的定义

24-变量04-输出

25-变量05-变量在函数中的作用域

26-变量06-变量在代码块中的作用域

### 27-变量07-练习-变量值的互换

交换两个变量的值

a = b – a;

b = b – a;

a = b + a;

### 28-scanf函数01-变量的内存分析

变量先定义, 内存地址越大

&: 地址运算符

%p: 输出地址 ( 例: int i = 1; printf(“&i: %p”, &i); )

### 29-scanf函数02-基本使用

scanf 只接收变量的地址( 例: int a; scanf(“%d”, &a); )

scanf 中不能写 \n

30-scanf函数03-练习

31-scanf函数04-其他用法

## 02\_基本运算

01-算术运算01-基本使用

02-算术运算02-使用细节

03-算术运算03-练习

04-赋值运算

05-自增自减运算01-基本使用

06-自增自减运算02-a++和++a的区别

07-自增自减运算03-补充

08-sizeof

09-关系运算01-基本使用

10-关系运算02-优先级

11-关系运算03-练习

### 12-逻辑运算01-简介

逻辑与: && 若左边不成立, 右边不计算

位与: & 两边都参与计算

13-逻辑运算02-逻辑与

14-逻辑运算03-逻辑或

15-逻辑运算04-逻辑非

16-三目运算01-基本使用

17-三目运算02-练习

## 03\_流程控制

### 01\_选择结构

#### 01-if01-第1种结构

if(10 > 0)

int a = 1; // 作用域不明确

02-if02-第2种结构

03-if03-第3种结构

04-if04-第4种结构

05-if05-使用注意

06-if06-练习

#### 07-switch01-基本使用

case后面定义变量必须加 {}

08-switch02-使用注意

09-switch03-练习

10-switch04-if和switch对比

### 02\_循环结构

01-while01-基本使用

02-while02-continue

03-while03-break

04-while04-小结

05-while05-练习01-简单实现

06-while06-练习01-代码完善

07-while07-练习02

08-while08-使用注意

09-do while

10-for01-基本使用

11-for02-使用注意

12-for03-循环嵌套

13-for04-练习

14-流程控制总结

15-continue和break使用注意

## 04\_函数

01-基本概念

02-基本使用

03-定义新的函数

04-函数的执行过程

05-形参和实参

06-返回值01-return的作用

07-返回值02-使用注意

08-练习

09-使用注意和函数声明

10-#include01-基本使用

11-#include02-stdio.h的作用

12-多文件团队开发01-基本实现

13-多文件团队开发02-完善代码

### 14-main函数和printf函数

一个中文占三个字节

printf 返回是字符串的字符数

## 05\_内存剖析

### 01-进制01-基本概念

二进制: 0b 或 0B

八进制: 0开头, 输出 %o

十六进制: 0x 或 0X, 输出 %x

02-进制02-代码编写

03-进制03-printf的多进制格式

04-进制04-变量在内存中的存储细节

05-进制05-练习

06-进制06-二进制转十进制

07-进制07-十进制转二进制

08-进制08-n位二进制的取值范围

09-进制09-负数和取值范围

### 10-类型说明符01-short和long

输出: %ld

### 11-类型说明符02-signed和unsigned

输出: %u

### 12-位运算01-按位与

交换两数

a = a ^ b;

b = a ^ b;

a = a ^ b;

13-位运算02-按位或

14-位运算03-按位异或

15-位运算04-按位取反

16-位运算05-左移

17-位运算06-右移

18-位运算07-练习01-变量值的互换

19-位运算08-练习02-判断奇偶性

20-位运算09-练习03-输出二进制位

21-char01-内存存储细节

22-char02-使用注意

23-char03-其他用法

24-char04-练习01

25-char05-练习02-小写转大写

## 06\_复杂数据类型

### 01\_数组

01-定义

02-简单使用01-赋值

03-简单使用02-遍历

#### 04-使用注意

错误1: 需要在定义的时定义个数 或 进行初始化

int ages[];

错误2: 只能在定义时赋值

int ids[];

ids = {1, 2}; // 数组名是首元素地址

错误3: 初始化赋值情况下, [] 内必须为常量或省略

int num = 9;

int ids[num] = {0, 1, 2};

05-计算数组长度

06-内存存储细节

#### 08-数组和函数

数组作为函数参数时, 可以省略元素个数,传递的是数组的地址

数组作为函数参数时, 当做指针变量使用, 指针变量占用八个字节

09-练习02

10-二维数组

### 02-字符串

01-基本使用

#### 02-内存存储细节

字符串以 0 结尾

‘\0’ 的ASCII 是0

strlen 计算字符串的字符个数 (一个汉字三个字符)

03-\0的作用

04-strlen

05-练习01-for循环实现

06-练习02-while循环实现

07-字符串数组

### 03-指针

#### 01-基本使用

指针变量存储地址

02-练习01

03-使用注意

04-指向指针的指针

07-练习03

08-疑问

#### 09-指针与数组

int ages[] = {1, 2, 3};

int \*p = ages;

int count = sizeof(ages) / sizeof(int);

for(int i = 0; I < count; i++) {

printf(“ages[%d]: %d - %d\n”, p[i], \*(p + i));

}

#### 10-指针与字符串

定义字符串的两种方式

1, char name[] = “it”; // 变量

2, char \*c = “it”; // 常量

11-指针数组和字符串输入

#### 13-返回指针的函数

char \* fun() {

return “yy”;

}

char \*name = fun();

#### 14-指向函数的指针

int sum(int a , int b) {

return a + b;

}

int (\*pSum)(int, int) = sum;

int i = pSum(1, 2); // 调用方式1

int j = (\*pSum)(2, 3); // 调用方式2

### 04-其他数据类型

01-全局变量和局部变量

#### 02-结构体01-基本使用

定义结构体第一种方式

struct Person {

int age;

double height;

};

struct Person p = {.height = 2.1, .age = 21}; // 声明赋值

int age = p.age; // 调用

第二种方式

struct Student {

int age;

} student; // 定义结构体时定义变量

第三种方式

struct {

int age;

} human; // 匿名方式, 不能复用

03-结构体02-内存分析

04-结构体03-变量的多种定义方式

05-结构体04-类型的作用域

#### 06-结构体05-结构体数组

sturct Student students[] = {

{1}, {2}, {3}

};

student[1].age = 20; // 只能通过这种方式修改

#### 07-结构体06-指向结构体的指针

struct Student \*studentP = &student;

studentP->age = 30; 或 (\*studentP).age = 31;

09-结构体08-嵌套定义

09-结构体08-嵌套定义

11-数据类型总结

## 07\_其它

### 01-宏定义01-不带参数的宏

所有的预处理指令都是以 # 开头

宏名一般用大写或以k 开头

取消宏定义 #undef

参数和结果都加()

#define kSum(v1, v2) ((v1) + (v2))

#define kSquares(a) ((a) \* (a))

02-宏定义02-带参数的宏

03-条件编译

04-文件包含

05-typedef01-基本使用

06-typedef02-使用注意

07-static和extern对函数的作用

08-static和extern对变量的作用

09-static对局部变量的作用

10-递归

# 04-Objective-C

## 01\_基本语法

01-语法概述

编译链接

cc 文件名.m –framework Foundation

02-第1个OC程序

03-第2个OC程序01

04-第2个OC程序02

05-第3个OC程序

06-第4个OC程序

07-BOOL

## 02\_面向对象

### 01\_思想

01-面向过程和面向对象

#### 02-类和对象

第一个字母必须大写

不能有下划线

03-如何设计类

04-类的设计练习

### 02\_类和对象

#### 01-类的声明和实现

声明

@interface Car : NSObject {

@public

int wheel;

}

- (void)run {

}

@end

实现

@implementation Car

@end

int main {

Car \*car = [Car new];

car->wheel = 4;

[car run];

}

02-对象的创建

03-成员变量的直接赋值

04-对象的行为

05-方法与成员变量01

06-方法与成员变量02

07-对象的简单内存分析

08-对象与函数参数

09-常见错误

10-方法与函数的区别

11-类的合理设计01-基本类型的属性

12-类的合理设计02-对象类型的属性

13-方法的声明和实现

14-练习

15-匿名对象

16-学习方法

17-iOS文档安装

18-作业点评

### 03\_三大特性

#### 01-封装01-set方法

@interface Student : NSObject {

int \_age;

}

@end

@implementation Student

- (void)setAge:(int)age {

\_age = age;

}

- (int)age {

return \_age;

}

@end

02-封装02-get方法

03-封装03-封装的好处

04-封装04-成员变量的命名规范

05-封装05-练习

06-弱语法

07-类方法01-基本使用

08-类方法02-与对象方法的区别

09-类方法03-练习

10-类方法04-使用注意

11-self01-访问成员变量

12-self02-调用方法

13-self03-使用注意

#### 14-继承01-基本使用

子父类不允许定义同名成员变量

15-继承02-继承的好处

16-继承03-使用注意

17-继承04-继承和组合

18-继承05-super

#### 19-多态01-基本使用

多态父类引用可以调用子类特有方法 (JAVA中会报错)

20-多态02-使用注意

21-多态03-多态的好处

22-多态04-局限性

### 04\_开发技巧

01-NSString01-基本使用

02-NSString02-学习方法

03-作业讲解01-点的设计

04-作业讲解02-点的测试

05-作业讲解03-圆的设计

06-作业讲解04-圆的内存分析

07-作业讲解05-圆的方法

08-多文件开发01-类的独立抽取

09-多文件开发02-.h和.m文件的分工

10-多文件开发03-使用Xcode新建类

11-Xcode功能演示01-新建类

12-Xcode功能演示02-断点调试

13-Xcode功能演示03-代码段保存

14-Xcode功能演示04-注释标记

15-Xcode功能演示05-多文件查看

### 05\_核心语法

01-点语法01-基本使用

02-点语法02-使用注意

#### 03-成员变量的作用域01-4种作用域

public

private

protected

04-成员变量的作用域02-使用注意和补充

#### 05-@property和@synthesize01-@property

property : 生成 set 和 get 方法的声明 (新特性可以同时生成声明和实现)

synthesize : 生成 set 和 get 方法的实现

06-@property和@synthesize02-@synthesize

07-@property和@synthesize03-setter和getter的最简写法

08-@property和@synthesize04-使用细节

09-id

10-构造方法01-基本概念

#### 11-构造方法02-重写init方法01

- (id)init() {

if(self = [super.init]) {

}

return self;

}

12-构造方法03-重写init方法02

13-构造方法04-init方法的执行过程

14-构造方法05-自定义构造方法01

15-构造方法06-自定义构造方法02

16-更改Xcode模板01-main.m

17-更改Xcode模板02-注释

18-分类01-基本使用

#### 19-分类02-使用注意

分类不能定义成员变量，可以访问原类中的成员变量

不建议分类覆盖原类中的成员方法

若原类和分类有相同的成员方法，最后编译的分类优先

20-分类03-给NSString增加类方法

21-分类04-给NSString扩充对象方法

22-作业点评

23-类的深入研究01-类的本质

24-类的深入研究02-类对象的使用

#### 25-类的深入研究03-类的加载和初始化

类加载完毕会调用 + (void)load

类初始化完毕会调用 +(void)initialize (若有分类只会加载分类的 initialize)

类描述 + (NSString \*)description

对象描述 - (NSString \*)description

26-description方法

27-NSLog输出补充

28-SEL01-基本使用

29-SEL02-其他用法

## 03-内存管理

### 01-为什么要管理内存

当调用 alloc/new/copy 引用计算器默认+1;

当调用 retain 引用计算器 +1;

当调用release 引用计算器 -1;

给野指针发送消息会报错, 空指针不会报错

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

[person release];

person.age = 10; // error

person = nil;

person.age = 10; // right

- (void)dealloc { // 当对象被回收会调用此方法

[super dealloc];

}

02-引用计数器01-简介

03-引用计数器02-retain、release的基本使用

04-引用计数器03-野指针和空指针

05-引用计数器04-计数器管理总结

06-多对象内存管理01-简介

07-多对象内存管理02-QQ堂实例

08-多对象内存管理03-代码演示01

09-多对象内存管理04-代码演示02

### 10-set方法内存管理01-基本实现

- (void)setCar:(Car \*)car {

if(\_car != car) {

[\_car release];

\_car = [car retain];

}

}

11-set方法内存管理02-代码完善

12-set方法内存管理03-练习加强

### 13-set方法内存管理04-@property的内存管理

@property (nonatomic, retain)Car \*car;

@property (nonatomic, assign, getter = isRich)BOOL rich;

14-set方法内存管理05-@property的参数

15-模型设计练习

### 16-循环retain和@class

### 17-autorelease01-基本使用

@autorelease {

Person \*person = [[[Person alloc] init] autorelease];

}

18-autorelease02-使用注意

19-autorelease03-常见错误

20-autorelease04-自动释放池的创建

21-autorelease05-实例01

22-autorelease06-实例02-使用规律

23-autorelease07-实例03-使用细节

24-内存管理小结

### 25-ARC01-基本原理

@property (nonatmic, strong) NSString \*name;

@Property (nonatmic, assign) int age;

26-ARC02-@property的strong和weak

### 27-ARC03-Xcode的ARC转换功能

1, 给某个文件添加 arc 或取消 arc

工程名 -> TARGETS -> Build Phases -> Compile Sources -> 双击某个文件

添加 arc : -f-objc-arc

取消 arc : -fno-objc-arc

2, 工程 arc 或 非arc

工程名 -> TARGETS -> Build Settings -> filter搜索 “arc” -> Apple LLVM 8.1 Language – Object C and ARC -> Object-C Automatic Reference Counting

### 28-ARC04-循环引用

## 04\_其它

### 01-block01-基本使用

int (^blockSum)(int, int) = ^ (int a, int b) {

return a + b;

}

typedef int (^BlockSum)(int, int);

02-block02-带参数和返回值的block

03-block03-使用typedef定义block类型

04-block04-总结

### 05-protocol01-基本使用

06-protocol02-@require、@optional、遵守多个协议

07-protocol03-协议间的遵守、基协议

08-protocol04-限制对象类型

09-protocol05-协议的提前声明

10-protocol06-总结

### 11-protocol07-代理设计模式

# 05-Foundation

01-结构体01-NSRange01-简介

02-结构体02-NSRange02-简单使用

03-结构体03-Point、Size、Rect01-简介

04-结构体04-Point、Size、Rect02-其他用法

05-字符串01-NSString

06-字符串02-NSMutableString

07-NSArray01-基本使用

08-NSArray02-遍历

09-综合练习01-计算单个文件的代码行数

10-综合练习02-计算整个文件夹的代码行数

11-NSMutableArray

12-NSSet

13-字典01-NSDictionary

14-字典02-NSMutableDictionary

15-字典03-遍历mp4

16-集合总结

17-NSNumber和NSValue

18-NSDate

进阶

# 01\_牛刀小试

00-开场白

01-iOS开发概述

02-搭建UI界面

03-初识UIViewController和UIView

04-处理按钮点击事件

05-IBAction和IBOutlet探究

06-初识程序启动过程

07-其他总结

08-开发环境简介

# 02-基础控件

## 01-UIView的常见属性

01-frame

02-bounds、center

03-tag

04-superview、subviews

05-transform简介

## 02-按钮操作

01-按钮样式和状态

02-常见错误

03-向上移动按钮

04-添加动画

05-其他方向移动按钮

06-代码重构

07-向左旋转按钮

08-放大按钮

09-右旋转、缩小按钮

10-再次重构

11-重置按钮状态

12-通过center移动按钮

13-小结

14-不同状态下的背景、文字、颜色

15-通过代码添加按钮

16-通过代码添加文本输入框

## 03-图片浏览器

01-基本界面

02-更换图片内容、序号

03-更换图片描述

04-BUG修正

05-显示和隐藏设置界面

06-开关控制夜间模式

07-控制图片缩放

08-作业说明

## 04-图片排列

01-添加UISegmentControl

02-简单排列图片

03-精确排列图片

04-动态更换图片列数

05-代码重构

## 05-汤姆猫

01-UIButton和UIImageView的区别

02-基本功能

03-其他功能

04-性能优化

## 06-联系人管理

01-UIToolbar的基本使用

02-添加新的一行

03-删除最后一行

04-增加动画效果

05-修改项目名称

06-添加头像和姓名

07-学习建议

08-精确获得某行的姓名

09-UIView常见方法总结

10-精确删除某一行

11-xib的简单使用

12-File's Owner01-基本使用

13-File's Owner02-详细解析

14-xib加载过程详解

15-File's Owner使用总结

16-view的封装01-自定义view

17-view的封装02-补充

# 03-高级控件

## 01-UIScrollView

01-大图片展示

02-其他属性简介

03-contentOffset

04-图片缩放01-功能实现

05-图片缩放02-原理解析

06-代理设计模式01

07-代理设计模式02

08-分页01-图片展示

09-分页02-显示页码

10-性能优化简介

11-键盘的简单处理01-快速清除按钮

12-键盘的简单处理02-退出键盘

13-键盘的简单处理03-键盘工具条

## 02-UITableView

### 01-多组数据展示

01-初识UITableView

02-简单展示

03-初探展示原理

04-扩充省份

05-用数组重构代码

06-显示头部尾部标题

07-用字典重构代码

08-用模型重构代码

09-索引条

10-Plain样式

### 02-单组数据展示

01-简单展示淘宝商品

02-用模型封装数据

03-弹框显示商品名称

04-刷新表格数据

05-局部刷新

## 03-性能优化

01-优化思路

02-代码实现

03-再次优化

04-搭建商品界面

05-展示商品数据

06-代码重构

07-标记选中某行

08-标记选中某行的过程分析

09-取消选中某行

10-显示选中的数量

11-删除选中的数据

12-其他思路实现删除数据

13-简化代码

## 04-编辑模式

01-删除数据

02-数据排序

03-cell内部结构

04-MVC模式