我不会，但我可以学

1.The Apple Worldwide Developers Conference(WWDC)

2. 苹果于WWDC 2015上宣布，Swift将开放源代码，包括编译器和标准库

3. Playground: a place where people can play

4. Swift 的多行注释可以嵌套在其他多行注释内部, 不要求在每行语句的结尾使用分号(;)，但当你在同一行书写多条语句时，必须用分号隔开

5. Swift中的字母采用的是Unicode编码[1]。Unicode叫做统一编码制，它包含了亚洲文字编码，如中文、日文、韩文等字符，甚至是我们在聊天工具中使用的表情符号

6. 如果一定要使用关键字作为标识符，可以在关键字前后添加重音符号（`），例如：let `class` = "Runoob"

7. Swift 空格:

eg: letl a= 1; (×) let a=1;(√) let a= 1+2;( ×) let a = 1 + 2;( √)

8. print 函数

public func print(\_ items: Any..., separator: String = default, terminator: String = default)

不换行输出,只需将最后一个参数赋值为空字符串即可:

eg: print("\(name)的官网地址为：\(site)")

9. 如果需要接收用户的输入可以使用 readLine()

let theInput = readLine()

10. swift可以自动识别属性类别。

11. 字符串是字符的序列集合(“abcd efg”),

可以通过使用字符串字面量或 String 类的实例来创建一个字符串：

eg: var stringA = "Hello, World!"

或 var stringB = String("Hello, World!")

判断字符串是否为空：stringA.isEmpty

字符串是否相等: stringA == stringB

长度：stringA .characters.count 或 stringA.count

转换字符串数字为整型 let myInt: Int? = Int(myString)

遍历字符串中的字符：for ch in "Runoob".characters {

print(ch)

}

字符串连接字符：varA.append( varB )

字符指的是单个字母(“a”): 型为 Character

Swift 中不能创建空的 Character（字符） 类型变量或常量

let char1: Character = "" (报错)

12. 类型别名: 对当前的类型定义了另一个名字， 使用 typealias 关键字 eg: typealias Feet = Int;var a:Feet = 10;

13. 如果你没有显式指定类型，Swift 会使用类型推断（type inference）来选择合适的类型。

14. Swift 是一个类型安全（type safe）的语言。 它会在编译你的代码时进行类型检查（type checks），并把不匹配的类型标记为错误。

**15. Swift 可选(Optionals)类型**

var a: Int? 或 var a: Optional<Int>

自动解析: 在声明可选变量时使用感叹号（!）替换问号（?）。这样可选变量在使用时就不需要再加一个感叹号（!）来获取值，它会自动解析(var myString:String!)

**16. Swift 字面量:** 就是指像特定的数字，字符串或者是布尔值这样，能够直接了当地指出自己的类型并为变量进行赋值的值.

1.25e2 表示 1.25 ⨉ 10^2

布尔值字面量有三个值:true false nil

**17. 位运算符**

&按位与： 同一位都为1时才为1

|按位或：任意一个为1就为1

^按位与或：同一位不同就为1，相同就为0

~按位取反：每一位都取反

<<按位左移：所有位向左移动指定的位数

>>按位右移：所有位向右移动指定的位数

**18. for 循环**在Swift 3 中已经弃用。使用for-in 循环

fallthrough 语句： 在一个case执行完后，继续执行下面的case

在大多数语言中，switch 语句块中，case 要紧跟 break，否则 case 之后的语句会顺序运行，而在 Swift 语言中，默认是不会执行下去的，switch 也会终止。如果你想在 Swift 中让 case 之后的语句会按顺序继续运行，则需要使用 fallthrough 语句。

repeat...while 循环：类似于do….while

**19. Swift 数组:** 有序列表相同类型的集合

创建数组：var someArray = [SomeType]()

var someArray:[Int] = [10, 20, 30]

var someInts = [Int](repeating: 10, count: 3)

(实例创建了一个类型为 Int ，数量为 3，初始值都为1 0 的空数组)

添加元素：append() 方法或 += 在数组末尾添加元素

someInts.append(20) ----添加元素

someInts += [40] ---- 添加数组

遍历数组：for item in someStrs

for (index, item) in someStrs.enumerated()

判断是否为空: intsA.isEmpty

**20. Swift 字典**: 无序列表的相同类型的集合（字典的key没有类型限制可以是整型或字符串，但必须是唯一的）。

创建字典：var someDict = [KeyType: ValueType]()

var someDict:[Int:String] = [1:"One", 2:"Two", 3:"Three"]

修改字典：updateValue(forKey:)

移除 Key-Value 对: removeValueForKey()

遍历字典: for (key, value) in someDict[value是key对应的值]

for (key, value) in someDict.enumerated()[value是key和value对]

字典转换为数组：let dictKeys = [Int](someDict.keys)

let dictValues = [String](someDict.values)

判断是否为空: intsA.isEmpty

**21.元组（tuple）**：与数组类似，不同的是，元组中的元素可以是任意类型，使用的是圆括号。

**22.函数：**func minMax(array: [Int]) -> (min: Int, max: Int) {}

函数参数名称:局部参数名(在函数的实现内部使用)

func sample(number: Int) {println(number) } ---number

外部参数名(局部参数名前指定外部参数名，中间以空格分隔， 用于在函数调用时传递给函数的参数)

func pow(firstArg a: Int, secondArg b: Int) -> Int {return res}

---- firstArg\ secondArg

注意：如果你提供了外部参数名，那么函数在被调用时，必须使用外部参数名。

可变参数：可变参数通过在变量类型名后面加入（...）

func methodName<N>(members: N...){

for i in members { print(i) }

}

变量参数（改变传入参数的值）：在参数定义前加 inout 关键字

func getName(\_ name: inout String).........

当传入的参数作为输入输出参数时，需要在参数名前加 & 符，表示这个值可以被函数修改。

Eg: func swapTwoInts(\_ a: inout Int, \_ b: inout Int) {

let temporaryA = a；a = b；b = temporaryA

}

var x = 1; var y = 5; swapTwoInts(&x, &y)

print("x 现在的值 \(x), y 现在的值 \(y)") ----x=5,y=1

**23. Swift 闭包**

{ (parameters) -> return type in

statements

}

尾随闭包: