# 学习语法之前的提醒

1. C语言属于一门高级语言，其实，所有高级语言的基本语法组成部分都是一样的，只是表现形式不太一样
2. 就好像亚洲人和非洲人，大家都有人类的结构：2只 手、2只脚、1个头，只是他们外表不太一样，比如肤色、脸型
3. 因此，你掌握好了一门高级语言，再去学习其他高级语言，那是相当快的
4. 而且，很多其他高级语言，比如后面要学习的Objective-C，都是基于C语言、从C语言衍生出来的，好好学习C语言吧，绝对不吃亏

# 关键字

## 什么是关键字

1. 关键字就是C语言提供的有特殊含义的符号，也叫做“保留字”
2. C语言一共提供了32个关键字，这些关键字都被C语言赋予了特殊含义

auto double int struct break else long switch

case enum register typedef char extern return union

const float short unsigned continue for signed void

default goto sizeof volatile do if while static

大致浏览一遍即可，不用去百度每个关键字的作用，这些关键字以后会经常用到的，到时候你想不记住都难

## 关键字的特征

1. 全部都是小写
2. 在开发工具或者智能文本编辑工具中会显示特殊颜色。默认情况下，C语言中的所有关键字在Xcode中都会显示紫褐色

## main函数中哪些是关键字

int、return

# 标识符

## 什么是标识符

标识符就是在程序中自定义的一些符号和名称。要跟关键字区分开来：关键字是C语言默认提供的符号，标识符是程序员自定义的

## 标识符的作用

1. 标识符，从字面上理解就是用来标识某些东西的符号，标识的目的就是为了将这些东西区分开来
2. 其实，标识符的作用就跟人类的名字差不多，为了区分每个人，就在每个人出生的时候起了个名字
3. C语言是由函数构成的，一个C程序中可能会有多个函数，为了区分这些函数，就给每一个函数都起了个名称。函数的名称就是标识符的一种。除了函数，以后还会学到“变量”这个概念，变量的名称也是标识符

## 命名

1. 命名规则（一定要遵守）

* 只能由26个英文字母的大小写、10个阿拉伯数字0~9、下划线\_组成
* 严格区分大小写，比如test和Test是2个不同的标识符
* 不能以数字开头
* 不可以使用关键字作为标识符

1. 命名规范（最好遵守）

* 尽量起个有意义的名称，比如一个完整的英文单词，别人一看这个名称就能才能这个标识符的作用。如果不懂英文，你也可以用拼音，尽量不要起像abcde、sfsdfsdf等这类看起来没有意义的名称
* 如果标识符中含有多个单词，可以使用驼峰标识（除开第一个单词，后面每个单词的首字母都是大写）：firstName、myFirstName，或者使用下划线\_来连接：first\_name、my\_first\_name

## 常见的标识符命名错误

|  |  |  |
| --- | --- | --- |
| 合法标识符 | 非法标识符 | 注释 |
| fromNo12 | from#12 | 标识符中不能使用#符号 |
| my\_Boolean | my-Boolean | 标识符中不能使用“-”符号，应使用下划线“\_”代替 |
| Obj2 | 2ndObj | 标识符不能使用数字开头 |
| myInt | int | “int”是内建关键字 |
| jack\_rose | jack&rose | 符号“&”不能出现在标识符中 |
| GUI | G.U.I | 标识符内部需要出现“.”分隔符 |

## 习题

指出标识符的正确性

test1 Mike2jack My\_tExt \_test

test!32 haha(da)tt 哈哈\_text 123haha

78text a \_123 \_

# 注释

## 什么是注释

1. 注释是在所有计算机语言中都非常重要的一个概念，从字面上看，就是注解、解释的意思
2. 注释可以用来解释某一段程序或者某一行代码是什么意思，方便程序员之间的交流。假如我写完一行代码后，加上相应的注释，那么别人看到这个注释就知道我的这行代码是做什么用的
3. 注释可以是任何文字，也就是说可以写中文
4. 在开发工具中注释一般是豆沙绿色

## 单行注释

1. 单行注释以两个正斜杠开头，也就是以//开头，只能注释一行，从//开始到这行的结尾都是注释的内容
2. 任何地方都可以写注释：函数外面、里面，每一条语句后面

## 多行注释

多行注释以/\*开头，以\*/结尾，/\*和\*/中间的内容都是注释

## 注释的作用

1. 被注释的代码不会参与编译

* 注释是写给人看的，不是给计算机看的。计算机怎么可能看得我们写的中文嘛。因此，当编译程序的时候，并不会将注释编译到.o目标文件中
* 从.o文件的大小可以间接看出注释后的代码并没有被编译

1. 检查代码的作用
2. 排除错误

## 注释的嵌套现象

1. 单行注释可以嵌套单行注释、多行注释

// 哇哈哈 // 呵呵呵

// /\* fsdfsdf \*/ // sdfsdfsd

1. 多行注释可以嵌套单行注释

/\*

// 作者：MJ

// 描述：第一个C语言程序

作用：这是一个主函数，C程序的入口点

\*/

1. 多行注释不能嵌套多行注释

/\* 哈哈哈

/\*嘻嘻嘻\*/

呵呵呵 \*/

1. 下面的写法是错误的

// /\*

哈哈哈

\*/

## 注释的重要性

* 要养成写注释的良好习惯。绝大部分项目经理检查下属代码的第一件事就是看有没有写注释，也有很多公司的机试也会检查注释（机试就是给你一道编程题、一台电脑，在规定时间内解题）
* 今天，你写了几百行代码，很高兴，做出了一个非常不错的功能，但是，忘了写注释。一个星期过后，你再回去看那一段代码，你可能完全看不懂了，这是很正常的事。如果你写了注释，那情况就不一样了，注释可以帮助你回顾代码的作用。
* 你在某家公司待了1年多，写了10几万行代码，但是你不写一点注释。有一天你离职了，新的员工接手你做的项目，他首先要做的事情肯定是要读懂你写 的代码。可是你一点注释都没写，10万行代码，全部都是英文，这会让这位新员工非常地蛋疼。每个人都有自己的思想，写代码的思路肯定是不一样的，看别人写 的代码是件非常痛苦的事情，特别是没有注释的代码。你不写注释的做法会大大降低公司的开发效率。因此，所有的正规公司都非常重视注释。

## 习题

看看下列程序能否运行成功，如果能，说出运行结果：

1. 程序1

// 主函数

int main() {

printf("itcast\n");

return 0;

}

1. 程序2

// 主函数

int main() {

printf("//itcast\n");

return 0;

}

1. 程序3

主函数

int main() {

printf("itcast\n");

return 0;

}

1. 程序4

int main()

{

printf(//"itcast\n");

return 0;

}

1. 程序5

int main() // 主函数 {

//printf("itcast\n");

return 0;

}