

# C语言 04 流程控制

---

## 选择

if

switch

## 循环

for

while

do..while

break 和 continue

## 什么是流程控制？

程序代码执行的顺序

## 如何看懂一个程序：

- 1.流程
- 2.每个语句的功能
- 3.试数

## 对于一些小算法的程序

- 》尝试中间编程解决它，大部分人无法自己解决
- 》解决不了，看答案，尝试看懂（重点）
- 》之后尝试自己去修改程序，并且指定不同输出结果的含义
- 》照着答案敲 / 不看答案，自己独立敲出来
- 》调试错误
- 》实在不行，就背下来

## 流程控制的分类

顺序

## 选择

定义

有些代码可能执行，可能不执行，有选择的执行某些代码

分类

## if

### i. if 最简单的用法

if (表达式)

需执行的语句；

### ii. if 的范围问题——不加 { }，只执行一个语句

对比

if (表达式)

语句A；

语句B；

if (表达式)

{

语句A；

语句B；

}

### iii. if。。。else 的用法

### iv. else if ....else if....的用法

✗错误示范

if (a > 0)

语句A；

语句B；

else if (a == 0)

语句C；

else

语句D；

原因分析——》由于if 只能控制一个语句，故语句B将整个 if。。else给隔开了，编译器无法识别 else if

v. if 空语句

```
if (XXXX);  
语句A;
```

## switch

格式

```
switch (表达式)  
{  
    case常量表达式1:  
        语句 ;  
        break;  
    case常量表达式2:  
        语句;  
        break;  
    case常量表达式3:  
        语句;  
        break;  
    .....  
    default:  
        语句;  
        break;  
}
```

## 循环

for

定义

某些代码会被重复执行

分类

## 1. 格式

for (初始化变量; 完成条件; 一次循环后执行的操作)

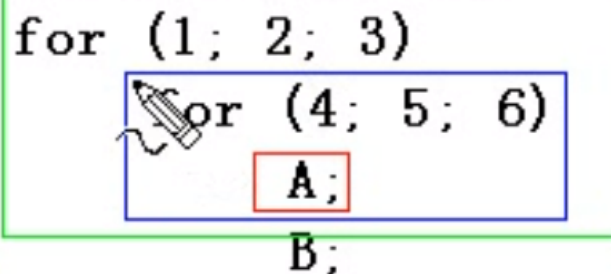
语句A;

**attention!**

由于电脑存储浮点型不精确 (5——》5.000001) , 循环中的初始化变量尽量少用浮点数

## 2. 执行的流程

### 多个for循环的嵌套使用



```
for (1; 2; 3)
    for (4; 5; 6)
        A;
    B;
```

1——》2——》【4——》5——》A——》5——》A....(循环多次, 直至5不成立)】——》6——》3——》2——》【4。。。 (继续循环)】

## while

### 1. 执行顺序

格式:

while (表达式)

语句;

### 2. 与for的相互比较

while 和 for 可以相互转换

但for逻辑性更强, 更不容易出错, 推荐多使用for

for (1; 2; 3)

语句A;

等价于

1;

while (条件)

{

    执行语句A;

    后续操作3;

}

3.什么时候用for，用while? ——》看语感

## do..while

格式

do

{

    表达式

} while (条件)

先做再判断

## break 和 continue

break

只在循环中起作用

在循环——》来终止循环

在 switch——》终止switch

在switch多层嵌套中——》break只能终止距离它最近的switch

在 if ——》不能直接用于if，除非if属于循环的一个子句

continue

只在循环中起作用

执行完该语句后，跳过本次循环余下的语句，直接跳转到是否需要再次执行循环的判断