

△选择&循环

2020年10月7日 20:34

• if语句

if (x) 等价于if (x!=0)

if (! x) 等价于if (x==0)

else就近匹配if/按照大括号，与缩进无关。

• switch语句

switch语句的一般形式：

```
switch(整形表达式)
{
    case 常量表达式1: 语句1;
    case 常量表达式2: 语句2;
    ...
    case 常量表达式n: 语句n;
    default: 语句n+1;
}
```

1. switch语句中使用的表达式必须具是int或enum类型，否则如float等其他数据类型是无法通过的编译的，因为编译器需要switch后面的语句和case后面的值精确匹配，而计算机无法精确表达一个float数据类型
2. switch可以任意个case语句(包括没有), case后面的语句不需要用括号括起来，每个case 后跟一个要比较的值和一个冒号。
3. case后面的值必须是int类型值，或者返回结果为int类型的表达式.
4. case后面的判断值必须是不同的值。
5. 当switch后面的变量值和case后面的常量值匹配相等后,case后面的代码将会被执行，直到break语句被执行后跳出switch代码块，所以case顺序不影响结果。
6. break不是必须的，如果没有break，则执行完当前case的代码块后会继续执行后面case代码块的内容，直到执行break才可以退出
7. switch有一个默认的情况，我们用default关键词表示，当switch后面的变量和所有case后面的常量都不匹配的情况下,默认执行default后面的语句。如果程序中没有default语句，程序则退出switch case结构。
8. switch语句不会在执行判断为真后的语句之后跳出循环，而是继续执行后面所有case语句。

Example

```
#include <stdio.h>

int main () {

    /* Local variable definition */
    char grade;
    scanf("%c", &grade);

    switch(grade) {
        case 'A' :
            printf("Excellent!\n" );
            break;
        case 'B' :
        case 'C' :
            printf("Well done\n" );
            break;
        case 'D' :
            printf("You passed\n" );
            break;
        case 'F' :
            printf("Better try again\n" );
            break;
        default :
            printf("Invalid grade\n" );
    }

    printf("Your grade is  %c\n", grade );
}
```

• for语句

计数控制的循环

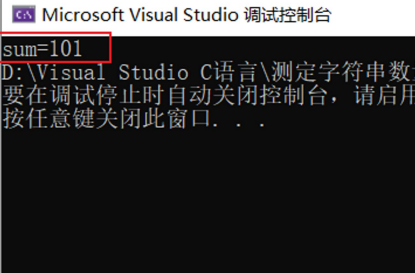
for (int i=0;i<10;i++) 在for括号内部可以定义i, **此i只能在for循环内部使用**, 占用地
址与外部i不同

for (;;) or for(1;) 此时为死循环, 永真条件, 需要用break来控制循环的结束

for(i=0;i<100;i++); or for(i=0;i<100;i++){ } 此时为空语句

使用时要注意:

```
#include<stdio.h>
int main()
{
    int i, sum=0;
    for (i = 1;i <= 100; i++);
    {
        sum = sum + i;
    }
    printf("sum=%d", sum);
    return ;
}
```



Microsoft Visual Studio 调试控制台
sum=101
D:\Visual Studio C语言\测定字符串数...
要在调试停止时自动关闭控制台, 请启用...
按任意键关闭此窗口...

for函数小括号内部使用两个;; 而不是, ,

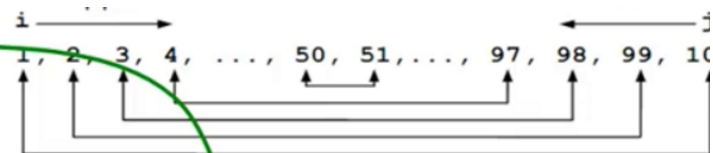
for函数第一行后面不要加;

如上图, 如果加了会输出101, 即每次循环执行一个空语句, 一直到循环结束, i取值为101的时候, 大括号里才执行一次。

逗号运算符 (Comma Operator)

表达式1, 表达式2, ..., 表达式n

- 多数情况下, 并不使用整个逗号表达式的值
- 更常见的是分别得到各表达式的值——顺序求值运算符
- 主要用在循环语句中, 同时对多个变量赋初值等



```
#include <stdio.h>
int main()
{
    int i, j, sum = 0;
    for (i=1,j=100; i<=j; i++,j--)
    {
        sum = sum + i + j;
    }
    printf("sum = %d", sum);
    return 0;
}
```

分号分隔的内部可以用, 来给多个变量赋值。

2.while语句

表达式1;

循环初始条件

while (表达式2)

循环控制条件

{

语句1

语句2

表达式3;

循环转化条件

}

当型循环结构 (先检票后上车)

先判断某些条件是否为真，然后再执行循环体。

do-while语句在while后面有;

while语句在while后面没有;

while (x) 如果x等于0，则会退出循环。

3.do-while语句

表达式1;

循环初始条件

do{

语句1

语句2

表达式3;

循环转化条件

}**while** (表达式2);

循环控制条件

直到型循环结构 (先上车后检票)

先执行了一次循环体之后，再对控制条件进行判断，

当条件不满足时执行循环体，满足时则停止。

至少执行一次

能用do-while循环描述的程序一定能用while循环和for 循环描述。

使用do-while语句的时候注意输出**n**还是**n-1**

• 辅助控制语句

break 语句

只能用于循环体语句和switch语句。

1.可以使流程跳出switch结构，继续执行switch下面的语句。

2.可以用来从循环体内跳出循环体，既结束当前循环，执行循环下面的语句。

注意：break语句只能跳出一层循环。

continue 语句

功能：结束本次循环，即跳过循环体尚未执行的语句，接着进行下一次是否执行循环的判定。

注意：continue语句结束本次循环

continue语句只是结束本次循环，而不是中止整个循环，接着进行下一次是否执行循环的判定。

break语句则是结束整个循环过程，不再判断执行循环的条件是否成立。

goto 语句和标号

goto语句又叫无条件转移语句。

形式：

goto 标号;

.....

标号: 语句; //一定要有冒号

Ø 功能：把程序控制转移到标号指定的语句处。执行goto语句后，程序从指定标号处的语句继续执行。

- 注意: goto语句常用的用法是用它退出多重循环。
- ! goto语句为非结构化语句，我们一般不提倡使用！

举例：

```
#include<stdio.h>

void main()
{
    int a=2,b=3;
    if(a<b)
        goto aa;
    printf("hello");
    aa:printf("s");
    return 0;
}
```