



知识点总结



- **条件语句**是C语言中用于根据不同条件，执行不同代码的控制结构。
- 常见的if条件语句有：
 - (1) **if语句**
 - (2) **if-else语句**
 - (3) **if-else if-else语句**（多级分支结构）
 - (4) **嵌套if语句**



(1) if语句：

当条件为真时，则执行if内的所有语句（即花括号里的语句）；

当条件为假时，则跳过if内的所有语句，继续执行后面的代码。

语法格式：

```
if (条件) {
```

```
    // 条件为真时，执行此处代码
```

```
    语句1;
```

```
    语句2;
```

```
    ...
```

```
}
```



(2) if-else语句:

当条件为真时，则执行if分支内的所有语句；
当条件为假时，则执行else分支内的所有语句。

语法格式:

```
if (条件) {  
    // 条件为真时，执行此处代码  
    语句1;  
    语句2;  
    ...  
}  
else {  
    // 条件为假时，执行此处代码  
    语句3;  
    语句4;  
    ...  
}
```



(3) if-else if-else语句（多级分支结构）：

如果条件1为真，就执行if分支内的所有语句；

如果条件1为假，程序会按从上到下的顺序，继续判断后续的else if分支的条件是否为真。如果某个分支的条件为真，就执行该分支内的所有语句。

如果前面所有条件都为假，则执行else分支内的所有语句。

语法格式：

```
if (条件1) {  
    // 条件1为真时，执行此处代码  
    语句1;  
    语句2;  
    ...  
}  
else if (条件2) {  
    // 条件2为真时，执行此处代码  
    语句3;  
    语句4;  
    ...  
}  
... // 可以有多个else if语句  
else {  
    // 前面所有分支的条件都为假时，执行此处代码  
    语句5;  
    语句6;  
    ...  
}
```



(4) 嵌套if语句:

在一个if语句的分支中，又包含了其它if条件语句。

只有当外层的条件为真时，才会判断内层的条件是否为真。

语法格式:

```
if (条件1) {  
    if (条件2) { // 条件1为真时，才判断条件2的真假  
        // 条件1和条件2都为真，执行此处代码  
        语句1;  
        语句2;  
        ...  
    }  
    else {  
        // 条件1为真，条件2为假时，执行此处代码  
        语句3;  
        语句4;  
        ...  
    }  
}
```



- 对条件进行判断时，通常有“真”和“假”两种状态。
- 在C语言中，可以用**布尔类型（bool）**来表示条件的真假，其中true值对应“真”，false值对应“假”。
- 使用布尔类型时，要先包含**<stdbool.h>**头文件。

例子：

```
#include <stdbool.h>
```

```
int main() {  
    bool flag = true;  
    if (flag) {  
        语句1;  
        ...  
    }  
}
```



- C语言中，也可以用**整数**来表示真假，通常用1表示“真”，0表示“假”。

例子：

```
int main() {  
    int flag = 1;  
    if (flag) {  
        语句1;  
        ...  
    }  
}
```

- 注意，在C语言里，任意**非零值**都会被视作条件为真。只有当值为0时，条件才为假。



- 在条件语句中，还可以结合关系运算符来形成**条件表达式**。
- 常见的**关系运算符**有：`==`（等于）、`!=`（不等于）、`>`（大于）、`<`（小于）、`>=`（大于等于）、`<=`（小于等于）。

例子：

```
int main() {  
    int flag = 1;  
    if (flag > 0) {  
        语句1;  
        ...  
    }  
}
```



·关于if条件语句还要注意以下几点：

1、if或else if或else语句下有**多条**执行语句时，必须用**一对花括号**把所有执行语句包裹起来。否则，该条件语句只会对紧跟在它后面的下一条语句起作用，而其他语句将始终被执行。

比如：

```
if (条件)
```

```
    语句1;
```

```
    语句2;
```

这里的if只能控制是否执行语句1，而语句2总是会被执行。



2、else总是与在其前面的，离它最近的，尚未配对的同层if配对。

比如：

```
if (条件1) {  
    if (条件2) {  
        语句1;  
    }  
    if (条件3) {  
        语句2;  
    }  
    else {          // 此else与同层的if(条件3)分支配对  
        语句3;  
    }  
}  
else {             // 此else与同层的if(条件1)分支配对  
    语句2;  
}
```



3、在if和else语句之间不能有其他非条件语句，否则会导致编译错误。

比如：这是个错误的例子。

```
if (条件) {
```

```
    语句1;
```

```
}
```

```
printf("err");
```

```
else {
```

```
    语句2;
```

```
}
```



else总是跟在它前面，离它最近的，尚未配对的同层if配对

题目1、为了避免在嵌套的if-else语句中产生二义性，C语言规定与else子句配对是（D）

- A. 与其在同一行上的if子句
- B. 在其之后最近的不带else的if子句
- C. 与其缩进位置相同的if子句
- D. 在其之前最近的不带else的同层if子句



题目2、下列条件语句中，输出结果与其他语句不同的是 (D)

假设: $a = 0$

A.

```
if (a)
    printf("%d\n", x);
else
    printf("%d\n", y);
```

C.

```
if (a != 0)
    printf("%d\n", x);
else
    printf("%d\n", y);
```

B.

```
if (a == 0)
    printf("%d\n", y);
else
    printf("%d\n", x);
```

D.

```
if (a == 0)
    printf("%d\n", x);
else
    printf("%d\n", y);
```



题目3、有以下程序段：

```
if (a > b)
    printf("x=%d,", x);
else
    printf("y=%d,", y);
if (a <= b)
    i++;
else
    j++;
```

若变量已正确定义并赋值，则与此程序段功能相同的选项是（ A ）

a > b：输出 x 且 j++

a <= b：输出 y 且 i++

A.

```
if (a > b) {
    printf("x=%d,", x);
    j++;
}
else {
    printf("y=%d,", y);
    i++;
}
```

C.

```
if (a <= b) {
    printf("x=%d,", x);
    i++;
}
else {
    printf("y=%d,", y);
    j++;
}
```

B.

```
if (a > b) {
    printf("x=%d,", x);
    i++;
}
else {
    printf("y=%d,", y);
    j++;
}
```

D.

```
if (a >= b) {
    printf("x=%d,", x);
    i++;
}
else {
    printf("y=%d,", y);
    j++;
}
```



C语言里，任意非零值都被当作真



题目4、有以下程序段：

```
#include <stdio.h>
main()
{
    int x = 0x13;
    if (x = 0x12)
        printf("True");
    printf("False\n");
}
```

程序运行后的输出结果是 (**B**)

A. True

B. TrueFalse

C. False

D. TrueFalseTrue



当不用一对花括号把if语句里的执行语句括起来时，if语句只能对紧跟着它的下一条语句起作用



题目5、若变量已正确定义，有以下程序段：

```
int a = 3, b = 5, c = 7;
if (a > b)
    a = b;
    c = a;
if (c != a)
    c = b;
printf("%d,%d,%d\n", a, b, c);
```

其输出结果是（ B ）

- A. 程序段有语法错误
- B. 3,5,3
- C. 3,5,5
- D. 3,5,7



题目6、有以下程序

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    int a = 0, b = 0, c = 0, d = 0;
```

```
    if (a = 1)
```

```
        b = 1;
```

```
        c = 2;
```

```
    else
```

```
        d=3;
```

```
    printf("%d,%d,%d,%d\n", a, b, c,  
d);
```

```
}
```

程序输出 (**B**)

A. 0,0,0,3

B. 编译有错

C. 1,1,2,0

D. 0,1,2,0



else必须跟if配对，它们之间不能存在其他非条件语句。



题目7、请编写一段代码，给定整数变量x（初始值为6）和y，要求用if语句实现下面的分段函数计算，并输出y值。

$$y = \begin{cases} 2x + 9 & (x > 10) \\ x + 1 & (1 \leq x \leq 10) \\ 0 & (x = 0) \\ 2x - 9 & (x < 0) \end{cases}$$

```
#include <stdio.h>

int main() {
    int x = 6;
    int y;

    if (x > 10) {
        y = 2 * x + 9;
    }
    else if (x >= 1) {
        y = x + 1;
    }
    else if (x == 0) {
        y = 0;
    }
    else {
        y = 2 * x - 9;
    }
    printf("y = %d\n", y);
}
```



题目8、请编写一个程序，根据学生的成绩输出相应的成绩等级。

成绩等级划分标准如下：

若成绩在90分以上（含90分）的等级为A；

若成绩在60~89分之间（含60分）的等级为B；

若成绩在60分以下的等级为C。

已知某学生成绩为70.5，请使用嵌套if语句来完成这个任务，并且要求将成绩变量score定义为double类型。

```
#include <stdio.h>

int main() {
    double score = 70.5;

    if (score < 60) {
        printf("C\n");
    }
    else {
        if (score >= 90) {
            printf("A\n");
        }
        else {
            printf("B\n");
        }
    }
}
```