C语言

# C 程序结构

C 程序主要包括以下部分：

预处理器指令；函数；变量；语句 & 表达式；注释

#include <stdio.h>

int main() { /\* 我的第一个 C 程序 \*/ printf("Hello, World! \n"); return 0; }

1. 程序的第一行 *#include <stdio.h>* 是预处理器指令，告诉 C 编译器在实际编译之前要包含 stdio.h 文件。
2. 下一行 *int main()* 是主函数，程序从这里开始执行。
3. 下一行 /\*...\*/ 将会被编译器忽略，这里放置程序的注释内容。它们被称为程序的注释。
4. 下一行 *printf(...)* 是 C 中另一个可用的函数，会在屏幕上显示消息 "Hello, World!"。
5. 下一行 **return 0;** 终止 main() 函数，并返回值 0。

**宏定义**

* 宏是通过 #define 指令定义的符号常量或代码片段。宏在编译前由预处理器替换为其定义的内容。常用于定义常量或者复杂的代码块。

**基本数据类型**  
它们是算术类型，包括整型（int）、字符型（char）、浮点型（float）和双精度浮点型（double）。

为了得到某个类型或某个变量在特定平台上的准确大小，可以使用 **sizeof** 运算符。表达式 *sizeof(type)* 得到对象或类型的存储字节大小。获取 int 类型的大小：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| float | 4 字节 | | 1.2E-38 到 3.4E+38 | | 6 位有效位 |
| double | 8 字节 | | 2.3E-308 到 1.7E+308 | | 15 位有效位 |
| long double | 16 字节 | | 3.4E-4932 到 1.1E+4932 | | 19 位有效位 |
| char | | 1 字节 | | -128 到 127 或 0 到 255 | |
| int | | 2 或 4 字节 | | -32,768 到 32,767 或 -2,147,483,648 到 2,147,483,647 | |

自动转换规则：

* a）浮点数赋给整型，该浮点数小数被舍去；
* b）整数赋给浮点型，数值不变，但是被存储到相应的浮点型变量中；

强制类型转换形式: **(类型说明符)(表达式)**

## 浮点常量

浮点常量由整数部分、小数点、小数部分和指数部分组成。您可以使用小数形式或者指数形式来表示浮点常量。

当使用小数形式表示时，必须包含整数部分、小数部分，或同时包含两者。当使用指数形式表示时， 必须包含小数点、指数，或同时包含两者。带符号的指数是用 e 或 E 引入的。

## 整数常量

整数常量可以是十进制、八进制或十六进制的常量。前缀指定基数：0x 或 0X 表示十六进制，0 表示八进制，不带前缀则默认表示十进制。

## 定义常量

使用 **#define** 预处理器： #define 可以在程序中定义一个常量，它在编译时会被替换为其对应的值。

### **const 关键字（不能更改）**

您可以使用 **const** 前缀声明指定类型的常量，如下所示：

const 数据类型 常量名 = 常量值;

#define 不进行类型检查，因为它只是进行简单的文本替换。而 const 定义的常量具有类型信息，编译器可以对其进行类型检查。这可以帮助捕获一些潜在的类型错误。

getchar（）

变量=getchar()

//接收键盘输入的一个字符

例

#include<stdio h>

main()

{char ch;

Ch=getchar(); //将键盘输入的一个字符存放到ch变量中.Ch);//将ch变量中的字符输出到屏幕printf( "%c\n"

/用于吸收上一次键盘输入数据后所带的分隔符(不存储)getchar();//作用等同于用%\*c

不回显的字符输入

ch=getch();//不回显也不用回车表示结束

## 算术运算符

下表显示了 C 语言支持的所有算术运算符。假设变量 **A** 的值为 10，变量 **B** 的值为 20，则：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| + | 把两个操作数相加 | A + B 将得到 30 |
| - | 从第一个操作数中减去第二个操作数 | A - B 将得到 -10 |
| \* | 把两个操作数相乘 | A \* B 将得到 200 |
| / | 分子除以分母 | B / A 将得到 2 |
| % | 取模运算符，整除后的余数 | B % A 将得到 0 |
| ++ | 自增运算符，整数值增加 1 | A++ 将得到 11 |
| -- | 自减运算符，整数值减少 1 | A-- 将得到 9 |

a++ 与 ++a 的区别：

先赋值后运算&&先运算后赋值

#include <stdio.h>

int main() {

int c; int a = 10; c = a++;

printf("先赋值后运算：\n");

printf("Line 1 - c 的值是 %d\n", c );

printf("Line 2 - a 的值是 %d\n", a );

a = 10; c = a--; printf("Line 3 - c 的值是 %d\n", c );

printf("Line 4 - a 的值是 %d\n", a );

printf("先运算后赋值：\n");

a = 10; c = ++a;

printf("Line 5 - c 的值是 %d\n", c );

printf("Line 6 - a 的值是 %d\n", a );

a = 10; c = --a; printf("Line 7 - c 的值是 %d\n", c );

printf("Line 8 - a 的值是 %d\n", a ); }

先赋值后运算：Line 1 - c 的值是 10Line 2 - a 的值是 11Line 3 - c 的值是 10Line 4 - a 的值是 9先运算后赋值：Line 5 - c 的值是 11Line 6 - a 的值是 11Line 7 - c 的值是 9Line 8 - a 的值是 9

## 关系运算符

|  |  |  |
| --- | --- | --- |
| == | 检查两个操作数的值是否相等，如果相等则条件为真。 | (A == B) 为假。 |
| != | 检查两个操作数的值是否相等，如果不相等则条件为真。 | (A != B) 为真。 |

## 逻辑运算符

下表显示了 C 语言支持的所有关系逻辑运算符。假设变量 **A** 的值为 1，变量 **B** 的值为 0，则：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| && | 称为逻辑与运算符。如果两个操作数都非零，则条件为真。 | (A && B) 为假。 |
| || | 称为逻辑或运算符。如果两个操作数中有任意一个非零，则条件为真。 | (A || B) 为真。 |
| ! | 称为逻辑非运算符。用来逆转操作数的逻辑状态。如果条件为真则逻辑非运算符将使其为假。 | !(A && B) 为真。 |

## 判断语句

C 语言提供了以下类型的判断语句。点击链接查看每个语句的细节。

|  |  |
| --- | --- |
| **语句** | **描述** |
| [if 语句](https://www.runoob.com/cprogramming/c-if.html" \o "C 中的 if 语句" \t "https://www.runoob.com/cprogramming/_blank) | 一个 **if 语句** 由一个布尔表达式后跟一个或多个语句组成。 |
| [if...else 语句](https://www.runoob.com/cprogramming/c-if-else.html" \o "C 中的 if...else 语句" \t "https://www.runoob.com/cprogramming/_blank) | 一个 **if 语句** 后可跟一个可选的 **else 语句**，else 语句在布尔表达式为假时执行。 |
| [嵌套 if 语句](https://www.runoob.com/cprogramming/c-nested-if.html" \o "C 中的嵌套 if 语句" \t "https://www.runoob.com/cprogramming/_blank) | 您可以在一个 **if** 或 **else if** 语句内使用另一个 **if** 或 **else if** 语句。 |
| [switch 语句](https://www.runoob.com/cprogramming/c-switch.html" \o "C 中的 switch 语句" \t "https://www.runoob.com/cprogramming/_blank) | 一个 **switch** 语句允许测试一个变量等于多个值时的情况。 |
| [嵌套 switch 语句](https://www.runoob.com/cprogramming/c-nested-switch.html" \o "C 中的嵌套 switch 语句" \t "https://www.runoob.com/cprogramming/_blank) | 您可以在一个 **switch** 语句内使用另一个 **switch**语句。 |

**switch语句一般形式：**

switch(表达式){

case 常量表达式1:语句1;

case 常量表达式2:语句2;

...

default:语句n+1;}

与 if 语句的不同：

if 语句中若判断为真则只执行这个判断后的语句，执行完就跳出 if 语句，不会执行其他 if 语句；而 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。在每一 case 语句之后增加 break 语句，使每一次执行之后均可跳出 switch 语句，从而避免输出不应有的结果。

## 循环类型

C 语言提供了以下几种循环类型。点击链接查看每个类型的细节。

|  |  |
| --- | --- |
| **循环类型** | **描述** |
| [while 循环](https://www.runoob.com/cprogramming/c-while-loop.html" \o "C 中的 while 循环) | 当给定条件为真时，重复语句或语句组。它会在执行循环主体之前测试条件。 |
| [for 循环](https://www.runoob.com/cprogramming/c-for-loop.html" \o "C 中的 for 循环) | 多次执行一个语句序列，简化管理循环变量的代码。 |
| [do...while 循环](https://www.runoob.com/cprogramming/c-do-while-loop.html" \o "C 中的 do...while 循环) | 除了它是在循环主体结尾测试条件外，其他与 while 语句类似。 |
| [嵌套循环](https://www.runoob.com/cprogramming/c-nested-loops.html" \o "C 中的嵌套循环) | 您可以在 while、for 或 do..while 循环内使用一个或多个循环。 |

[C while 和 do while 区别](https://www.runoob.com/w3cnote/c-while-and-do-while.html" \t "https://www.runoob.com/cprogramming/_blank)

while 循环适用于在循环之前检查条件，而 do while 循环适用于至少执行一次循环体，不管条件是否成立。

break 语句与 continue 语句的区别，前者是结束整个循环过程，后者这是结束本次循环。

while语句

#include <stdio.h>  
int main (){  
   int a = 10;  
   while( a < 20 )   {   printf("a 的值： %d**\n**", a);    a++;  }

return 0;}

for语句

(循环起始条件;循环条件;循环变量增值)

Eox (initial value ; condition; update counter)

## 无限循环

for( ; ; ) { printf("该循环会永远执行下去！\n"); }

您可以按 **Ctrl + C** 键终止一个无限循环。

## 随机数

## 库函数的使用----随机数的产生

## 伪随机

## rand()%100+0

## //0-99rand(%个数+下限[0,100);0-2147483647(默认)

## int magic;

## srand(time(NULL));/\*为函数rand设置随机种子，time(NULL)返回值被转换为一个无符号的整数，可以作为随机种子\*/

## magic= rand()%101+1;//[1,101)1-100

## 初始化数组

在 C 中，您可以逐个初始化数组，也可以使用一个初始化语句，如下所示：

double balance[5] = {1000.0, 2.0, 3.4, 7.0, 50.0};

大括号 { } 之间的值的数目不能大于我们在数组声明时在方括号 [ ] 中指定的元素数目。

如果省略掉了数组的大小，数组的大小则为初始化时元素的个数。因此，如果：

double balance[] = {1000.0, 2.0, 3.4, 7.0, 50.0};

拓展

一．

system(“cls")://清屏

system(“pause"):冻结屏幕,暂停效果

system(“color 0A”);///改变控制台的背景色0和前景色A(黑底绿字),系统默认的设置是0F (黑底白字).各颜色代码如下(不区分大小写):

二．

延时

Sleep()延时，单位:毫秒在windowsh(大写S)P422(stdlib.h有误)Sleep(1000)//延时1000毫秒，即延时1秒