

## C 函数

函数是一组一起执行一个任务的语句。每个 C 程序都至少有一个函数，即主函数 **main()**，所有简单的程序都可以定义其他额外的函数。

您可以把代码划分到不同的函数中。如何划分代码到不同的函数中是由您来决定的，但在逻辑上，划分通常是根据每个函数执行一个特定的任务来进行的。

函数**声明**告诉编译器函数的名称、返回类型和参数。函数**定义**提供了函数的实际主体。

C 标准库提供了大量的程序可以调用的内置函数。例如，函数 **strcat()** 用来连接两个字符串，函数 **memcpy()** 用来复制内存到另一个位置。

函数还有很多叫法，比如方法、子例程或程序，等等。

## 定义函数

C 语言中的函数定义的一般形式如下：

```
return_type function_name( parameter list )
{
    body of the function
}
```

在 C 语言中，函数由一个函数头和一个函数主体组成。下面列出一个函数的所有组成部分：

- **返回类型**：一个函数可以返回一个值。**return\_type** 是函数返回的值的数据类型。有些函数执行所需的操作而不返回值，在这种情况下，**return\_type** 是关键字 **void**。
- **函数名称**：这是函数的实际名称。函数名和参数列表一起构成了函数签名。
- **参数**：参数就像是占位符。当函数被调用时，您向参数传递一个值，这个值被称为实际参数。参数列表包括函数参数的类型、顺序、数量。参数是可选的，也就是说，函数可能不包含参数。
- **函数主体**：函数主体包含一组定义函数执行任务的语句。

## 实例

以下是 **max()** 函数的源代码。该函数有两个参数 **num1** 和 **num2**，会返回这两个数中较大的那个数：

```
/* 函数返回两个数中较大的那个数 */
int max(int num1, int num2)
{
    /* 局部变量声明 */
    int result;
    if (num1 > num2)
        result = num1;
    else
        result = num2;
```

```
return result;
}
```

## 函数声明

函数**声明**会告诉编译器函数名称及如何调用函数。函数的实际主体可以单独定义。

函数声明包括以下几个部分：

```
return_type function_name( parameter list );
```

针对上面定义的函数 max(), 以下是函数声明：

```
int max(int num1, int num2);
```

在函数声明中，参数的名称并不重要，只有参数的类型是必需的，因此下面也是有效的声明：

```
int max(int, int);
```

当您在源文件中定义函数且在另一个文件中调用函数时，函数声明是必需的。在这种情况下，您应该在调用函数的文件顶部声明函数。

## 调用函数

创建 C 函数时，会定义函数做什么，然后通过调用函数来完成已定义的任务。

当程序调用函数时，程序控制权会转移给被调用的函数。被调用的函数执行已定义的任务，当函数的返回语句被执行时，或到达函数的结束括号时，会把程序控制权交还给主程序。

调用函数时，传递所需参数，如果函数返回一个值，则可以存储返回值。例如：

### 实例

```
#include <stdio.h>
/* 函数声明 */
int max(int num1, int num2);
int main ()
{
    /* 局部变量定义 */
    int a = 100;
    int b = 200;
    int ret;
    /* 调用函数来获取最大值 */
    ret = max(a, b);
    printf( "Max value is : %d\n", ret );
    return 0;
}
/* 函数返回两个数中较大的那个数 */
int max(int num1, int num2)
{
    /* 局部变量声明 */
```

```
int result;
if (num1 > num2)
result = num1;
else
result = num2;
return result;
}
```

把 max() 函数和 main() 函数放一块，编译源代码。当运行最后的可执行文件时，会产生下列结果：

```
Max value is : 200
```

## 函数参数


如果函数要使用参数，则必须声明接受参数值的变量。这些变量称为函数的**形式参数**。


形式参数就像函数内的其他局部变量，在进入函数时被创建，退出函数时被销毁。

当调用函数时，有两种向函数传递参数的方式：

调用类型	描述
<a href="#">传值调用</a>	该方法把参数的实际值复制给函数的形式参数。在这种情况下，修改函数内的形式参数不会影响实际参数。
<a href="#">引用调用</a>	通过指针传递方式，形参为指向实参地址的指针，当对形参的指向操作时，就相当于对实参本身进行的操作。

默认情况下，C 使用**传值调用**来传递参数。一般来说，这意味着函数内的代码不能改变用于调用函数的实际参数。

 9 篇笔记

 写笔记