本小节内容

全局变量解析-形参-实参解析
局部变量与全局变量

1 全局变量解析-形参-实参解析

在不同的函数之间传递数据时，可以使用的方法如下：

（1）参数：通过形式参数和实际参数。

（2）返回值：用 return 语句返回计算结果。

（3）全局变量：外部变量。

下面来看一个全局变量的实例，如例 1.1 所示。

【例 1.1】全局变量的使用。

```
#include <stdio.h>

int i=10;  //全局变量
void print(int a)
{
    printf("print i=%d\n",i);
}
int main()
{
    {
        int j=5;
    }  //局部变量的有效范围是离自己最近的花括号
    printf("main i=%d\n",i);
    i=5;
    print(i);
    return 0;
}
```

全局变量存储在哪里？如图 1.1 所示，全局变量 i 存储在数据段，所以 main 函数和 print 函数都是可见的。全局变量不会因为某个函数执行结束而消失，在整个进程的执行过程中始终有效，因此工作中应尽量避免使用全局变量！在前几章中，我们在函数内定义的变量都称为局部变量，局部变量存储在自己的函数对应的栈空间内，函数执行结束后，函数内的局部变量所分配的空间将会得到释放。**如果局部变量与全局变量重名，那么将采取就近原则，即实际获取和修改的值是局部变量的值。**

图 1.1 全局变量的存储

练习：如果把 print(int a)改为 print(int i)，那么 print 函数的打印结果会是多少?

关于形参与实参的一些说明如下。

(1) 定义函数中指定的形参，如果没有函数调用，那么它们并不占用内存中的存储单元。只有在发生函数调用时，函数 print 中的形参才被分配内存单元。在调用结束后，形参所占的内存单元也会被释放。

(2) 实参可以是常量、变量或表达式，但要求它们有确定的值，例如，print(i+3)在调用时将实参的值 i+3 赋给形参。print 函数可以有两个形参，如 print(int a,int b)

(3) 在被定义的函数中，必须指定形参的类型。如果实参列表中包含多个实参，那么各参数间用逗号隔开。**实参与形参的个数应相等**，类型应匹配，且实参与形参应按顺序对应，一一传递数据。

(4) 实参与形参的类型应相同或赋值应兼容。

(5) 实参向形参的数据传递是单向“值传递”，只能由实参传给形参，而不能由形参传回给实参。在调用函数时，给形参分配存储单元，并将实参对应的值传递给形参，调用结束后，形参单元被释放，实参单元仍保留并维持原值。

(6) 形参相当于局部变量，因此不能再定义局部变量与形参同名，否则会造成编译不通。

2 局部变量与全局变量

1. 内部变量

在一个函数内部定义的变量称为内部变量。它只在本函数范围内有效，即只有在本函数内才能使用这些变量，故也称局部变量。

关于局部变量需要注意如下几点:

(1) 主函数中定义的变量只在主函数中有效，而不因为在主函数中定义而在整个文件或程序中有效。主函数也不能使用其他函数中定义的变量。

(2) 不同函数中可以使用相同名字的变量，它们代表不同的对象，互不干扰。

(3) 形式参数也是局部变量。

(4) 在一个函数内部，可以在复合语句中定义变量，这些变量只在本复合语句中有效，这种复合语句也称“分程序”或“程序块”。例 1.1 中的 int j=5 就是如此，只在离自己最近的花括号内有效，**若离开花括号，则在其下面使用该变量会造成编译不通。**

(5)注意一个细节，for 循环的小括号内定义的 int i，在离开 for 循环后，是不可以再次使用的。

2. 外部变量

函数之外定义的变量称为外部变量。外部变量可以为本文件中的其他函数共用，它的有效范围是从定义变量的位置开始到本源文件结束，所以也称全程变量。

关于全局变量需要注意如下几点：

(1) 全局变量在程序的全部执行过程中都占用存储单元，而不是仅在需要时才开辟单元。

(2) 使用全局变量过多会降低程序的清晰性。在各个函数执行时都可能改变外部变量的值，程序容易出错，因此要有限制地使用全局变量(初试时尽量不用)。

(3) 因为函数在执行时依赖于其所在的外部变量，如果将一个函数移到另一个文件中，那么还要将有关的外部变量及其值一起移过去。然而，如果该外部变量与其他文件的变量同名，那么就会出现问题，即会降低程序的可靠性和通用性。**C 语言一般要求把程序中的函数做成一个封闭体，除可以通过“实参→形参”的渠道与外界发生联系外，没有其他渠道。**