

教程目录

1 编程基础

2 C语言初探

3 变量和数据类型

4 输入输出

5 分支结构和循环结构

6 C语言数组

7 C语言函数

8 预处理命令

9 C语言指针

9.1 1分钟彻底理解指针的概念

9.2 大话C语言指针变量

9.3 C语言指针变量的运算

9.4 数组指针（指向数组的指针）

9.5 字符串指针（指向字符串的指针）

9.6 C语言数组灵活多变的访问形式

9.7 指针变量作为函数参数

9.8 用C语言指针作为函数返回值

9.9 二级指针（指向指针的指针）

9.10 空指针NULL以及void指针

9.11 注意，数组和指针绝不等价

9.12 数组在什么时候会转换为指针

9.13 指针数组（每个元素都是指针）

9.14 一道题目教你玩转指针数组

9.15 指针与二维数组

9.16 函数指针（指向函数的指针）

9.17 只需一招，彻底攻克C语言指针

9.18 用main()函数接收控制台数据

9.19 对C语言指针的总结

10 结构体、位运算以及其他

11 文件操作

12 C语言调试

C语言指针变量作为函数参数

<上一节 下一节> 关注我们： 微信公众号 新浪微博 QQ交流群：588321099

C语言中文网推出辅导班啦，包括「C语言辅导班、C++辅导班、算法/数据结构辅导班」，全部都是一对一教学：一对一辅导 + 一对一答疑 + 布置作业 + 项目实践 + 永久学习。QQ在线，随时响应！

在C语言中，函数的参数不仅可以是整数、小数、字符等具体的数据，还可以是指向它们的指针。用指针变量作函数参数可以将函数外部的地址传递到函数内部，使得在函数内部可以操作函数外部的数据，并且这些数据不会随着函数的结束而被销毁。

像数组、字符串、动态分配的内存等都是一系列数据的集合，没有办法通过一个参数全部传入函数内部，只能传递它们的指针，在函数内部通过指针来影响这些数据集合。

有的时候，对于整数、小数、字符等基本类型数据的操作也必须要借助指针，一个典型的例子就是交换两个变量的值。

有些初学者可能会使用下面的方法来交换两个变量的值：

```
01. #include <stdio.h>
02.
03. void swap(int a, int b){
04.     int temp; //临时变量
05.     temp = a;
06.     a = b;
07.     b = temp;
08. }
09.
10. int main(){
11.     int a = 66, b = 99;
12.     swap(a, b);
13.     printf("a = %d, b = %d\n", a, b);
14.     return 0;
15. }
```

运行结果：

a = 66, b = 99

从结果可以看出，a、b的值并没有发生改变，交换失败。这是因为 swap() 函数内部的 a、b 和 main() 函数内部的 a、b 是不同的变量，占用不同的内存，它们除了名字一样，没有其他任何关系，swap() 交换的是它内部 a、b 的值，不会影响它外部（main() 内部）a、b 的值。

改用指针变量作参数后就很容易解决上面的问题：

```
01. #include <stdio.h>
02.
03. void swap(int *p1, int *p2){
04.     int temp; //临时变量
05.     temp = *p1;
06.     *p1 = *p2;
07.     *p2 = temp;
08. }
09.
10. int main(){
11.     int a = 66, b = 99;
12.     swap(&a, &b);
```

```
13.     printf("a = %d, b = %d\n", a, b);
14.     return 0;
15. }
```

运行结果：

a = 99, b = 66

调用 swap() 函数时，将变量 a、b 的地址分别赋值给 p1、p2，这样 *p1、*p2 代表的就是变量 a、b 本身，交换 *p1、*p2 的值也就是交换 a、b 的值。函数运行结束后虽然会将 p1、p2 销毁，但它对外部 a、b 造成的影响是“持久化”的，不会随着函数的结束而“恢复原样”。

需要注意的是临时变量 temp，它的作用特别重要，因为执行 `*p1 = *p2;` 语句后 a 的值会被 b 的值覆盖，如果不先将 a 的值保存起来以后就找不到了。

用数组作函数参数

数组是一系列数据的集合，无法通过参数将它们一次性传递到函数内部，如果希望在函数内部操作数组，必须传递数组指针。下面的例子定义了一个函数 max()，用来查找数组中值最大的元素：

```
01. #include <stdio.h>
02.
03. int max(int *intArr, int len){
04.     int i, maxValue = intArr[0]; //假设第0个元素是最大值
05.     for(i=1; i<len; i++){
06.         if(maxValue < intArr[i]){
07.             maxValue = intArr[i];
08.         }
09.     }
10.
11.     return maxValue;
12. }
13.
14. int main(){
15.     int nums[6], i;
16.     int len = sizeof(nums)/sizeof(int);
17.     //读取用户输入的数据并赋值给数组元素
18.     for(i=0; i<len; i++){
19.         scanf("%d", nums+i);
20.     }
21.     printf("Max value is %d!\n", max(nums, len));
22.
23.     return 0;
24. }
```

[纯文本](#) [复制](#)

运行结果：

12 55 30 8 93 27 ✓

Max value is 93!

参数 intArr 仅仅是一个数组指针，在函数内部无法通过这个指针获得数组长度，必须将数组长度作为函数参数传递到函数内部。数组 nums 的每个元素都是整数，scanf() 在读取用户输入的整数时，要求给出存储它的内存的地址，`nums+i` 就是第 i 个数组元素的地址。

用数组做函数参数时，参数也能够以“真正”的数组形式给出。例如对于上面的 max() 函数，它的参数可以写成下面的形式：

```
01. int max(int intArr[6], int len){
02.     int i, maxValue = intArr[0]; //假设第0个元素是最大值
03.     for(i=1; i<len; i++){
04.         if(maxValue < intArr[i]){
05.             maxValue = intArr[i];
06.         }
07.     }
08.     return maxValue;
09. }
```

`int intArr[6]` 好像定义了一个拥有 6 个元素的数组，调用 max() 时可以将数组的所有元素“一股脑”传递进来。

读者也可以省略数组长度，把形参简写为下面的形式：

```
01. int max(int intArr[], int len){
02.     int i, maxValue = intArr[0]; //假设第0个元素是最大值
```

```
03.     for(i=1; i<len; i++){
04.         if(maxValue < intArr[i]){
05.             maxValue = intArr[i];
06.         }
07.     }
08.     return maxValue;
09. }
```

`int intArr[]` 虽然定义了一个数组，但没有指定数组长度，好像可以接受任意长度的数组。

实际上这两种形式的数组定义都是假象，不管是 `int intArr[6]` 还是 `int intArr[]` 都不会创建一个数组出来，编译器也不会为它们分配内存，实际的数组是不存在的，它们最终还是会转换为 `int *intArr` 这样的指针。这意味着，两种形式都不能将数组的所有元素“一股脑”传递进来，大家还得规规矩矩使用数组指针。

`int intArr[6]` 这种形式只能说明函数期望用户传递的数组有 6 个元素，并不意味着数组只能有 6 个元素，真正传递的数组可以有少于或多于 6 个的元素。

需要强调的是，不管使用哪种方式传递数组，都不能在函数内部求得数组长度，因为 `intArr` 仅仅是一个指针，而不是真正的数组，所以必须要额外增加一个参数来传递数组长度。

C语言为什么不允许直接传递数组的所有元素，而必须传递数组指针呢？

参数的传递本质上是一次赋值的过程，赋值就是对内存进行拷贝。所谓内存拷贝，是指将一块内存上的数据复制到另一块内存上。

对于像 `int`、`float`、`char` 等基本类型的数据，它们占用的内存往往只有几个字节，对它们进行内存拷贝非常快。而数组是一系列数据的集合，数据的数量没有限制，可能很少，也可能成千上万，对它们进行内存拷贝有可能是一个漫长的过程，会严重拖慢程序的效率，为了防止技艺不佳的程序员写出低效的代码，C语言没有从语法上支持数据集合的直接赋值。

除了C语言，C++、Java、Python 等其它语言也禁止对大块内存进行拷贝，在底层都使用类似指针的方式来实现。

C语言中文网推出辅导班啦，包括「C语言辅导班、C++辅导班、算法/数据结构辅导班」，全部都是一对一教学：一对一辅导 + 一对一答疑 + 布置作业 + 项目实践 + 永久学习。QQ在线，随时响应！

编程帮，一个分享编程知识的公众号。跟着站长一起学习，每天都有进步。

通俗易懂，深入浅出，一篇文章只讲一个知识点。

文章不深奥，不需要钻研，在公交、在地铁、在厕所都可以阅读，随时随地涨姿势。

文章不涉及代码，不烧脑细胞，人人都可以学习。

当你决定关注「编程帮」，你已然超越了90%的程序员！



微信扫描二维码关注

< 上一节

下一节 >

关注我们：

微信公众号

新浪微博

QQ交流群：588321099

关于C语言中文网 | 关于站长 | 如何才能完成一部教程 | 联系我们 | 网站地图 | 手机版网站

精美而实用的网站，关注编程技术，追求极致，让您轻松愉快的学习。

Copyright ©2011-2015 biancheng.net, All Rights Reserved, 陕ICP备15000209号

biancheng.net