


数组名不等于指针---sizeof()函数求数组大小错误问题

 **C语言学习** 专栏收录该内容

90 订阅 15 篇文章 订阅专栏

前言：今天项目中需要求 采样Q点的数量并且遍历，采样点用数组存储，自定义了一个函数想用sizeof求其长度，然后遍历，结果失败了，查阅之后发现以下问题：

在main函数中，sizeof是可以正常工作的

```
1 | #include <stdio.h>
2 |
3 | int Number[10];
4 |
5 | int main()
6 | {
7 |     int size = sizeof(Number);
8 |     printf("数组大小为: %d\n",size);
9 |     int len = sizeof(Number)/sizeof(int);
10 |    printf("数组共有%d个数据\n",len);
11 |
12 |    return 0;
13 | }
```

输出：

```
数组大小为: 40
数组共有10个数据

-----
Process exited after 0.05772 seconds with return value 0
请按任意键继续. . .
```

但是在自定义函数中就不可以了，如下：

```
1 | #include <stdio.h>
2 |
3 | int Number[10];
4 |
5 | void print_1(int n[])
6 | {
7 |     int size = sizeof(n);
8 |     printf("数组大小为: %d\n",size);
9 |     int len = sizeof(n)/sizeof(int);
10 |    printf("数组共有%d个数据\n",len);
11 | }
12 |
13 |
14 | int main()
15 | {
16 |
17 |     print_1(Number);
18 |
19 |     return 0;
20 | }
```

```
数组大小为: 8
数组共有2个数据

-----
Process exited after 0.05344 seconds with return value 0
请按任意键继续. . .
```

那么我们首先要知道sizeof函数的功能：

sizeof是获取数据在内存中所占用的存储空间，以字节为单位来计数。

那么这个时候有的同学就会有问题了，两次传入的都是数组的首地址，为什么主函数中就可以，自定义函数中就不行呢？

得益于谭老爷子的C语言书籍普及量和销量，很多人认为数组名就是指向数组首地址的一个指针，但其实这个说法是错误的！

我们用一个最简单的例子，假设数组名是一个指针，那么：

```
1 | #include <stdio.h>
2 |
3 | int Number[10];
4 |
5 | int *Number2;
6 |
7 | int main()
8 | {
9 |     int a=sizeof(Number);
10 |    int b=sizeof(Number2);
11 |
12 |    printf("a的大小为: %d \n b的大小为 %d\n",a,b);
13 |
14 |    return 0;
15 | }
```

Number是一个指针，Number2也是一个指针，正常情况下 大小都应该为8 但是实际的输出确实 a=40 b=8

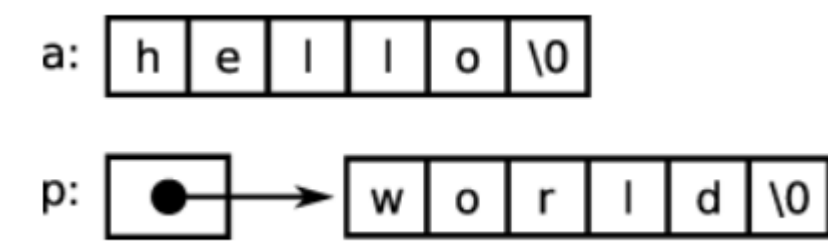
也就是说数组名在某些情况下是不等于指针的，只是在一些情况下会退化为指针

首先我们要知道，单纯的数组名，不是指针

数组名是一个标识符，它标识出我们之前申请的一连串内存空间，而且这个空间内的元素类型是相同的——即数组名代表的是一个内存块及这个内存块中的元素类型。

只是在大多数情况下数组名会“退化”（C标准使用的decay和converted这两个词）为指向第一个元素的指针。而指针不是一种聚合类的数据结构，它保存着某一种类型的对象的地址（void“除外”），也说它指向这个对象。我们可以通过这个地址访问这个对象。用一个图来解释，其中a代表了整个我们声明的内存块，p仅仅指向了一个char类型的对象

```
1 | char a[] = { 'h', 'e', 'l', 'l', 'o', '\0' };
2 | char b[] = { 'w', 'o', 'r', 'l', 'd' };
3 |
4 | char *p=b;
```



这是怎么回事呢？

我们看一下C99标准：

C99 6.3.2.1 Lvalues, arrays, and function designators 中第三段是这样说的：

Except when it is the operand of the sizeof operator or the unary & operator, or is a string literal used to initialize an array, an expression that has type "array of type" is converted to an expression with type "pointer to type" that points to the initial element of the array object and is not an lvalue. If the array object has register storage class, the behavior is undefined.

这段话的意思是：数组名只有在

1. sizeof 运算符
2. 取址 & 运算符
3. 字符常量初始化的数组 Str[]="abcdef"

这三种情况下不会发生退化(array decay)

其余情况下调用数组名，都会退化成指向数组首地址的指针

再深入的话，就是要了解指针的sizeof

指针是用来记录另一个对象的地址，所以指针的内存大小就等于计算机内部地址总线的宽度。

对一个地址来取大小呢，如果是32位系统的话即为4，如果是64位系统的话为8，所以呢，在函数中sizeof获取的是指针的长度而不是数组的长度

指针变量的sizeof值与指针所指的对象没有任何关系。

结论：
也就是说在c语言中，数组名在函数的调用中退化成了一个指针，对函数的参数使用Sizeof，sizeof获取的结果就是指针的大小，而不是数组本身的大小

再了解一下Sizeof的处理时间

sizeof是C语言的一种单目操作符（但有人也不这么以为，认为它是一种特殊的宏），如C语言的其他操作符++，-等一样。它并不是函数。sizeof操作符以字节形式给出了其操作数的存储大小。操作数可以是一个表达式或括在括号内的类型名。操作数的存储大小由操作数的类型决定，简单的说其作用就是返回一个对象或者类型所占的内存字节数。

也就是说，Sizeof是一个C语言的操作符，那么他的处理阶段在编译阶段，也就是说你程序没有运行前，sizeof(arr)就被替换成了一个固定的常量，那么对于动态生成的数组大小是不能用sizeof来算出来的。

解决方法：

在函数中多加一个参数，表示数组的长度，

```
1 | #include <stdio.h>
2 |
3 | int Number[10];
4 |
5 | void print_1(int n[], int len)
6 | {
7 |
8 |     printf("数组大小为: %d\n",len);
9 |
10 |    printf("数组共有%d个数据\n",len,sizeof(int));
11 |
12 | }
13 |
14 | int main()
15 | {
16 |
17 |    print_1(Number,sizeof(Number));
18 |
19 |    return 0;
20 | }
```

以上就是问题的总结。



点个赞再走吧！



📖 文章知识点与官方知识档案匹配，可进一步学习相关知识

算法技能树 > leetcode-数组 > 59-螺旋矩阵 II 5938 人正在系统学习中

解析sizeof, strlen, 指针以及数组作为函数参数的应用 01-01
代码如下所示： 代码如下 typedef struct st_test { int id; char *pName; char class[10];}Student;v...

C/C++数组名与指针区别 peterpanh的专栏 973
引言 指针是C/C++语言的特色，而数组名与指针有太多的相似，甚至很多时候，数组名可...

参与评论

请发表有价值的评论。 博客评论不欢迎灌水，良好的社区氛围需要大家一起来维护。 [去评论](#)

函数中使用sizeof(arr) / sizeof(arr[0])求数组长度不... 3-10
错误原因 我们可以使用sizeof(arr) / sizeof(arr[0])求数组长度,但是要注意,sizeof()函数是求数组...

数组名不完全等于指针_yilin54的博客 4-27
指针是C/C++语言的特色,而数组名与指针有太多的相似,甚至很多时候,数组名可以作为指针使...

sizeof()函数求类型所占字节大小-指针，数组 dragon_cdu4的博客 3344
举例说明： char*p; char test[10]; p=test; sizeof(p)=4(32位系统) //实质是求指针类型所占字节...

c语言 使用指针操作数组踩坑,sizeof(数组名) sizeof (指针... qq_38032878的博客 192
#include <stdio.h> #include <string.h> #include <stdlib.h> typedef struct test { int s[4]; char a[2]...

sizeof(数组名)和sizeof(指针)易混_sssupersly的博客 3-9
sizeof(p)获取的是指针地址,strlen(p)可以获取数组长度。 数组传递时也类似,虽然看上去printf...

C语言 sizeof数组名,别混淆了sizeof(数组名)和sizeof... 2-24
其实,函数printf参看上去像是一个数组,于是有的朋友就会认为它就是一个数组,于是就发生了...

数组类型的参数做函数入参，会退化为指针_陈顺发布 Oliver King 的小窝 240
数组类型的参数做函数入参，会退化为指针。函数内，再通过sizeof求大小，就是指针的大小了...

数组作为函数的参数时，不能通过sizeof运算符得到该数组的... zhghong的专栏 3369
当把数组作为函数的参数时，你无法在程序运行时通过数组参数本身告诉函数该数组的大小，...

有关sizeof 数组名的问题 数组名和指针的深入理解(C++... 3-21
标准C库函数strcpy的函数原型中能推断的两个参数都为char型指针,而我们在调用中传给它的却...

数组与指针的sizeof大小_feike24的博客_sizeof指针大小 3-20
数组与指针的sizeof大小 1 sizeof运算符,以字节为单位给出数据大小 strlen()函数,以...

C++ 安全编码：变量 Sunrise的专栏 2506
变量 指针变量、表示资源描述符的变量、BOOL变量声明必须赋予初值 为了贯彻零开销原则 (...

C++ sizeof 使用规则及陷阱分析 天下无双 884
C++ sizeof 使用规则及陷阱分析2008-09-07 06:00作者：出处：blog责任编辑：方舟 1、什...

C++ sizeof用法 ljreput的专栏 630
sizeof操作符的作用是返回一个对象或类型名的长度，长度的单位是字节。返回值的类型是标...

【C 语言】数组（指针退化验证）计算...让学习 成为一种习惯（韩耀亮 の 技术博客） 259
一、 指针退化验证、 二、 完整代码示例 一、 指针数组 11_指针数组的应用场景01

数组名和指针的深入理解（C++） weixin_30386713的博客 101
指针是C/C++语言的特色，而数组名与指针有太多的相似，甚至很多时候，数组名可以作为指...

关于数组和指针的sizeof问题 qq_43069548的博客 137
在C/C++中，当我们声明一个数组时，数组的名字也是一个指针，该指针指向数组的第一个元...

为什么数组传入函数后的sizeof不对了 枯树无花 1260
因为数组作为形参传入函数后会退化或指针，对一个地址来取大小呢，如果是32位系统的话即...

sizeof() && 数组名和指针 531
sizeof 函数计算时，对于数组起始地址和指向数组的指针，结果是不同的 #include int love(char...

sizeof()为什么不能得到指针指向内容的大小_陈门推荐 more_HH 1万+
首先，我们看看sizeof是什么？是一个操作符,也是关键字,就不是一个函数,这和strlen()不同...

《华为C&C++语言安全规范》笔记 2672
《华为C&C++语言安全规范》笔记 通过阅读《华为C&C++语言安全规范》1，我了解到我在...

c中自定义函数通过sizeof来输出数组的长度为何不正确？ ... 一直加班的程序猿 5892
这两天，在学习C语言的时候遇到一个bug，后来就在segmentfault提问，通过网友的回答也就...

“相关推荐”对你有帮助么？

非常没帮助 没帮助 一般 有帮助 非常有帮助

©2022 CSDN 皮肤主题：编程工作室 设计师：CSDN官方博客 返回首页

关于我 招贤纳士 商务合作 寻求报道 400-000-0108 kefu@csdn.net 在线客服 工作时间 8:30-22:00
公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1030-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 家长监护 网络110报警服务 中国互联网举报中心 Chrome商店下载 ©1999-2022北京创新乐知网络技术有限公司 版权与免责声明 版权申诉 出版物许可证 营业执照



举报