

High一下!

酷壳 - CoolShell

享受编程和技术所带来的快乐 - Coding Your Ambition
(<https://coolshell.cn/>)

C语言的谜题

📅 2009年05月31日 (<https://Coolshell.cn/Articles/945.Html>) 👤 陈皓
(<https://Coolshell.cn/Articles/Author/Haoel>) 🗣️ 112,277 人阅读

这几天，本站推出了几篇关于C语言的很多文章如下所示：

- 语言的歧义 [酷壳链接 (<https://coolshell.cn/articles/830.html>)] [CSDN 链接 (<http://blog.csdn.net/haoel/archive/2009/05/18/4197010.aspx>)]
- 谁说C语言很简单？ [酷壳链接 (<https://coolshell.cn/articles/873.html>)] [CSDN 链接 (<http://blog.csdn.net/haoel/archive/2009/05/26/4217950.aspx>)]
- 6个变态的C语言Hello World程序 [酷壳链接 (<https://coolshell.cn/articles/914.html>)] [CSDN 链接 (<http://blog.csdn.net/haoel/archive/2009/05/26/4217565.aspx>)]
- 如何加密 / 弄乱C源代码 [酷壳链接 (<https://coolshell.cn/articles/933.html>)] [CSDN 链接 (<http://blog.csdn.net/haoel/archive/2009/05/30/4225974.aspx>)]
- C语言的谜题 [酷壳链接 (<https://coolshell.cn/articles/945.html>)] [CSDN 链接 (<http://blog.csdn.net/haoel/archive/2009/06/01/4231029.aspx>)]

我们可以看到很多C语言相关的一些东西。比如《语言的歧义》主要告诉了大家C语言中你意想不到的错误以及一些歧义上的东西。而《谁说C语言很简单》则通过一些看似你从来不可能写出的代码来告诉大家C语言并不是一件容易事情。《6个变态的hello world》和《如何弄乱C的源代码》则以一种极端的方式告诉大家，不要以为咱们自己写不出混乱的代码，每个程序员其实都有把代码搞得一团乱的潜质。通过这些文章，相信你对编程或是你觉得很简单的C语言有了一些了解。是的，很不容易吧，以前是不是低估了编程和C语言？今天是否我们又在低估C++和Java呢？

本篇文章《C语言的谜题》展示了14个C语言的谜题以及答案，代码应该是足够清楚的，而且我也相信有相当的一些例子可能是我们日常工作可能会见到的。通过这些谜题，希望你能更了解C语言。如果你不看答案，不知道是否有把握回答各个谜题？让我们来试试。

1、下面的程序并不见得会输出 **hello-std-out**，你知道为什么吗？

```
1 #include <stdio.h>
2 #include <unistd.h>
3 int main()
4 {
5     while(1)
6     {
7         fprintf(stdout,"hello-std-out");
8         fprintf(stderr,"hello-std-err");
9         sleep(1);
10    }
11    return 0;
12 }
```

参考答案：stdout和stderr是不是同设备描述符。stdout是块设备，stderr则不是。对于块设备，只有当下面几种情况下才会被输入，1) 遇到回车，2) 缓冲区满，3) flush被调用。而stderr则不会。

2、下面的程序看起来是正常的，使用了一个逗号表达式来做初始化。可惜这段程序是有问题的。你知道为什么呢？

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int a = 1,2;
6     printf("a : %d\n",a);
7     return 0;
```

```
8 | }
```

参考答案：这个程序会得到编译出错（语法出错），逗号表达式是没错，可是在初始化和变量声明时，逗号并不是逗号表达式的意义。这点要区分，要修改上面这个程序，你需要加上括号：int a = (1,2);

3、下面的程序会有什么样的输出呢？

```
1 | #include <stdio.h>
2 | int main()
3 | {
4 |     int i=43;
5 |     printf("%d\n",printf("%d",printf("%d",i)));
6 |     return 0;
7 | }
```

参考答案：程序会输出4321，你知道为什么吗？要知道为什么，你需要知道printf的返回值是什么。printf返回值是输出的字符数。

4、下面的程序会输出什么？

```
1 | #include <stdio.h>
2 | int main()
3 | {
4 |     float a = 12.5;
5 |     printf("%d\n", a);
6 |     printf("%d\n", (int)a);
7 |     printf("%d\n", *(int *)&a);
8 |     return 0;
9 | }
```

参考答案：

该项程序输出如下所示，

```
0
12
1095237632
```

原因是：浮点数是4个字节，12.5f 转成二进制是：01000001010010000000000000000000，十六进制是：0x41480000，十进制是：1095237632。所以，第二和第三个输出相信大家也知道是为什么了。而对于第一个，为什么会输出0，我们需要了解一下float和double的内存布局，如下：

- **float**: 1位符号位(s)、8位指数(e), 23位尾数(m,共32位)
- **double**: 1位符号位(s)、11位指数(e), 52位尾数(m,共64位)

然后，我们还需要了解一下printf由于类型不匹配，所以，会把float直接转成double，注意，12.5的float和double的内存二进制完全不一样。别忘了在x86芯片下使用是反字节序，高位字节和低位字节要反过来。所以：

- **float版**: 0x41480000 (在内存中是：00 00 48 41)
- **double版**: 0x4029000000000000 (在内存中是：00 00 00 00 00 00 29 40)

而我们的%d要求是一个4字节的int，对于double的内存布局，我们可以看到前四个字节是00，所以输出自然是0了。

这个示例向我们说明printf并不是类型安全的，这就是为什么C++要引如cout的原因了。

5、下面，我们再来看一个交叉编译的事情，下面的两个文件可以编译通过吗？如果可以通过，结果是什么？

file1.c

```
1 | int arr[80];
```

file2.c

```
1 | extern int *arr;
2 | int main()
3 | {
4 |     arr[1] = 100;
5 |     printf("%d\n", arr[1]);
6 |     return 0;
7 | }
```

参考答案：该程序可以编译通过，但运行时会出错。为什么呢？原因是，在另一个文件中用 extern int *arr来外部声明一个数组并不能得到实际的期望值，因为他们的类型并不匹配。所以导致指针实际并没有指向那个数组。注意：一个指向数组的指针，并不等于一个数组。修改：extern int arr[]。（参考：ISO C语言 6.5.4.2 节）

6、请说出下面的程序输出是多少？并解释为什么？（注意，该程序并不会输出“b is 20”）

```
1 | #include <stdio.h>
2 | int main()
```

```

3  {
4      int a=1;
5      switch(a)
6      {
7          int b=20;
8          case 1:
9              printf("b is %d\n",b);
10             break;
11             default:
12                 printf("b is %d\n",b);
13                 break;
14         }
15         return 0;
16     }

```

参考答案：该程序在编译时，可能会出现一条warning: unreachable code at beginning of switch statement。我们以为进入switch后，变量b会被初始化，其实并不然，因为switch-case语句会把变量b的初始化直接就跳过了。所以，程序会输出一个随机的内存值。

7、请问下面的程序会有什么潜在的危险？

```

1  #include <stdio.h>
2  int main()
3  {
4      char str[80];
5      printf("Enter the string:");
6      scanf("%s",str);
7      printf("You entered:%s\n",str);
8      return 0;
9  }

```

参考答案：本题很简单了。这个程序的潜在问题是，如果用户输入了超过80个长度的字符，那么就会有数组越界的问题了，你的程序很有可能会crash了。

8、请问下面的程序输出什么？

```

1  #include <stdio.h>
2  int main()
3  {
4      int i;
5      i = 10;
6      printf("i : %d\n",i);
7      printf("sizeof(i++) is: %d\n",sizeof(i++));
8      printf("i : %d\n",i);
9      return 0;
10 }

```

参考答案：如果你觉得输出分别是，10，4，11，那么你就错了，错在了第三个，第一个是10没有什么问题，第二个是4，也没有什么问题，因为是32位机上一个int有4个字节。但是第三个为什么输出的不是11呢？居然还是10？原因是，sizeof不是一个函数，是一个操作符，其求i++的类型的size，这是一件可以在程序运行前（编译时）完全的事情，所以，sizeof(i++)直接就被4给取代了，在运行时也就不会再有了i++这个表达式。

9、请问下面的程序的输出值是什么？

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define SIZEOF(arr) (sizeof(arr)/sizeof(arr[0]))
5  #define PrintInt(expr) printf("%s:%d\n",#expr,(expr))
6
7  int main()
8  {
9      /* The powers of 10 */
10     int pot[] = {
11         0001,
12         0010,
13         0100,
14         1000
15     };
16
17     int i;
18     for(i=0;i<SIZEOF(pot);i++)
19         PrintInt(pot[i]);
20
21     return 0;
22 }

```

参考答案：好吧，如果你对于PrintInt这个宏有问题的话，你可以去看一看《语言的歧义 (<https://coolshell.cn/articles/830.html>)》中的第四个示例。不过，本例的问题不在这里，本例的输出会是：1，8，64，1000，其实很简单了，以C/C++中，以0开头的数字都是八进制的。

10、请问下面的程序输出是什么？（绝对不是10）

```
#include
#define PrintInt(expr) printf("%s : %dn",#expr,(expr))

int main()
{
    int y = 100;
    int *p;
    p = malloc(sizeof(int));
    *p = 10;
    y = y/*p; /*dividing y by *p */;
    PrintInt(y);
    return 0;
}
```

参考答案：本题输出的是100。为什么呢？问题就出在 $y = y/*p$ 上了，我们本来想的是 $y / (*p)$ ，然而，我们没有加入空格和括号，结果 $y/*p$ 中的 $/*$ 被解释成了注释的开始。于是，这也是整个恶梦的开始。

11、下面的输出是什么？

```
1  #include <stdio.h>
2  int main()
3  {
4      int i = 6;
5      if( ((++i < 7) && (i++/6)) || (++i <= 9))
6          ;
7
8      printf("%d\n",i);
9      return 0;
10 }
```

参考答案：本题并不简单的考前缀++或后缀++，本题主要考的是&&和||的短路求值的问题。所为短路求值：对于（条件1 && 条件2），如果“条件1”是false，那“条件2”的表达式会被忽略了。对于（条件1 || 条件2），如果“条件1”为true，而“条件2”的表达式则被忽略了。所以，我相信你会知道本题的答案是什么了。

12、下面的C程序是合法的吗？如果是，那么输出是什么？

```
1  #include <stdio.h>
2  int main()
3  {
4      int a=3, b = 5;
5
6      printf(&a["Ya!Hello! how is this? %s\n"], &b["junk/super"]);
7
8      printf(&a["WHAT%c%c%c %c%c %c !\n"], 1["this"],
9          2["beauty"],0["tool"],0["is"],3["sensitive"],4["CCCCC"]);
10
11     return 0;
12 }
```

参考答案：

本例是合法的，输出如下：

```
Hello! how is this? super
That is C !
```

本例主要展示了一种另类的用法。下面的两种用法是相同的：

```
"hello"[2]
2["hello"]
```

如果你知道： $a[i]$ 其实就是 $*(a+i)$ 也就是 $*(i+a)$ ，所以如果写成 $i[a]$ 应该也不难理解了。

13、请问下面的程序输出什么？（假设：输入 Hello, World）


```
1  #include <stdio.h>
2  int main()
3  {
4      char dummy[80];
5      printf("Enter a string:\n");
6      scanf("%[^\n]", dummy);
7      printf("%s\n", dummy);
8      return 0;
9  }
```


参考答案：本例的输出是“Hello, Wo”，scanf中的“ $^\n$ ”是作梗的东西。意思是遇到字符r就结束了。

14、下面的程序试图使用“位操作”来完成“乘5”的操作，不过这个程序中有个BUG，你知道是什么吗？

```
1  #include <stdio.h>
2  #define PrintInt(expr) printf("%s : %d\n",#expr,(expr))
3  int FiveTimes(int a)
4  {
5      int t;
6      t = a<<2 + a;
7      return t;
8  }
9
10 int main()
11 {
12     int a = 1, b = 2, c = 3;
13     PrintInt(FiveTimes(a));
14     PrintInt(FiveTimes(b));
15     PrintInt(FiveTimes(c));
16     return 0;
17 }
```

参考答案：本题的问题在于函数FiveTimes中的表达式“t = a<<2 + a;”，对于a<<2这个位操作，优先级要比加法要低，所以这个表达式就成了“t = a << (2+a)”，于是我们就得不到我们想要的值。该程序修正如下：[c] int FiveTimes(int a) { int t; t = (a<<2) + a; return t; } [/c] (全文完)

 C/C++ 语言
(<https://Coolshell.cn/Category/Proglanguage/Cplusplus>), 编程语言
(<https://Coolshell.cn/Category/Proglanguage>)

 C++ (<https://Coolshell.cn/Tag/C>)

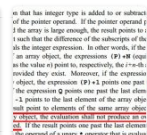
相关文章



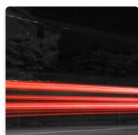
(<https://coolshell.cn/articles/11307.html>)



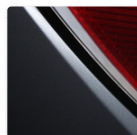
C语言结构体里的成员数组和指针产生的问题 (<https://coolshell.cn/articles/11466.html>)



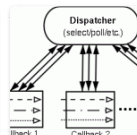
C语言的整型溢出问题 (<https://coolshell.cn/articles/11466.html>)



Leetcode 编程训练 (<https://coolshell.cn/articles/11466.html>)



一个“蝇量级”C语言 (<https://coolshell.cn/articles/11466.html>)



State Threads 回调终结者 (<https://coolshell.cn/articles/11466.html>)

《C语言的谜题》的相关评论



我哎程序员 (<http://www.52coder.net>)说道：

2016年06月01日 09:11 (<https://coolshell.cn/articles/945.html/comment-page-4#comment-1840535>)
第6题，为何我这输出b是20 windows gcc

Pingback: 如何学好C语言 - linux运维部落 (<http://www.178linux.com/2141>)

Pingback: C语言结构体里的成员数组和指针 - lol (<http://yang.lol/2017/01/06/c-%e8%af%ad%e8%a8%80%e7%bb%93%e6%9e%84%e4%bd%93%e9%87%8c%e7%9a%84%e6%88%90%e5%91%98%e6%95%b0%e>)

Pingback: 如何学好C语言 | Scott-Blog (<https://blog-scottwang.rhcloud.com/post/2017/05/21/how-to-study-c-well.html>)

Pingback: 如何学好C语言 - iszhou (<http://www.iszhou.com/2017/07/03/%e5%a6%82%e4%bd%95%e5%ad%a6%e5%a5%bd%e8%af%ad%e8%a8%80/>)



wuduer说道:

2017年10月23日 09:45 (<https://coolshell.cn/articles/945.html/comment-page-4#comment-1919527>)

12有错误

```
printf(&a["WHAT%c%c%c %c%c %c !\n"], 1["this"],
```

第一个参数不支持这种形式了。



wuduer说道:

2017年10月23日 09:54 (<https://coolshell.cn/articles/945.html/comment-page-4#comment-1919528>)

不好意思 看错了 这个没问题

Pingback: 一个fork的面试题 | 酷壳 - CoolShell (<https://coolshell.cn/articles/7965.html>)



wenshangzhongdu@163.com说道:

2018年12月25日 15:36 (<https://coolshell.cn/articles/945.html/comment-page-4#comment-1949193>)

这些谜题, 其实都不算什么, 也算不上谜题, 有的是c语言本身语法有缺陷, 比如 `y/*p`, 这个就是注释的语法有缺陷, 放到ide里看就很清楚了. 有的是linux操作系统的问题, 比如块设备和字符设备的缓冲, 不是c语言的问题.

还有的题目有点谭浩强的意思 `++i++`, 这本身就是不应该出现再代码里的. 还有的是运算符优先级的问题, 总之我感觉这些问题本身只能说明c语言有缺陷, 而不能说明c语言有多强.

Pingback: 当printf("-")遇上fork() —某公司招聘笔试题目 - 平头哥个人博客 (<http://pintouge.online/?p=616>)