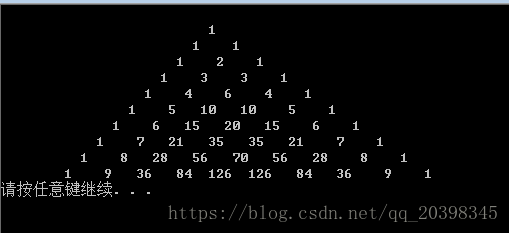
**C/C++常见笔试题1  
1、C和C++有什么不同?**  
【参考答案】  
从机制上：C是面向过程的（但C也可以编写面向对象的程序）；C++是面向对象的，提供了类。但是，C++编写面向对象的程序比C容易。  
从适用的方向：C适合要求代码体积小的，效率高的场合，如嵌入式；C++适合更上层的，复杂的; llinux核心大部分是C写的，因为它是系统软件，效率要求极高。  
从名称上也可以看出，C++比C多了+，说明C++是C的超集；那为什么不叫C+而叫C++呢，是因为C++比C来说扩充的东西太多了，所以就在C后面放上两个+；于是就成了C++。  
C语言是结构化编程语言，C++是面向对象编程语言。 C++侧重于对象而不是过程，侧重于类的设计而不是逻辑的设计。  
**2、“引用”与指针的区别是什么？**  
【参考答案】  
指针通过某个指针变量指向一个对象后，对它所指向的变量间接操作。程序中使用指针，程序的可读性差；  
而引用本身就是目标变量的别名，对引用的操作就是对目标变量的操作。  
1) 引用必须被初始化，指针不必。  
2) 引用初始化以后不能被改变，指针可以改变所指的对象。  
3) 不存在指向空值的引用，但是存在指向空值的指针  
**3、VC 中，编译工具条内的 Debug 与 Release 选项是什么含义？**  
【参考答案】  
Debug 通常称为调试版本，它包含调试信息，并且不作任何优化，便于程序员调试程序。  
Release 称为发布版本，它往往是进行了各种优化，使得程序在代码大小和运行速度上都是最优的，以便用户很好地使用。  
Debug 带有大量的调试代码，运行时需要相应的运行库，发布模式程序紧凑不含有调试代码和信息，直接可以运行（如果不需要运行库）  
**4、有了 malloc/free 为什么还要 new/delete  ？**  
【参考答案】  
malloc 与 free 是 C++/C 语言的标准库函数，new/delete 是 C++的运算符。它们都可用于申请动态内存和释放内存。   
对于非内部数据类型的对象而言，光用 maloc/free 无法满足动态对象的要求。对象在创建的同时要自动执行构造函数，对象在消亡之前要自动执行析构函数。  
由于malloc/free 是库函数而不是运算符，不在编译器控制权限之内，不能够把执行构造函数和析构函数的任务强加于malloc/free。  
因此 C++语言需要一个能完成动态内存分配和初始化工作的运算符 new，以及一个能完成清理与释放内存工作的运算符 delete。  
**5、内存的分配方式有几种？**  
【参考答案】  
一、从静态存储区域分配。内存在程序编译的时候就已经分配好，这块内存在程序的整个运行期间都存在。例如全局变量。  
二、在栈上创建。在执行函数时，函数内局部变量的存储单元都可以在栈上创建，函数执行结束时这些存储单元自动被释放。栈内存分配运算内置于处理器的指令集中，效率很高，但是分配的内存容量有限。  
三、从堆上分配，亦称动态内存分配。程序在运行的时候用malloc或new申请任意多少的内存，程序员自己负责在何时用free或delete释放内存。动态内存的生存期由我们决定，使用非常灵活，但问题也最多。  
**7、float a,b,c , 问等式 (a+b)+c==(b+a)+c 和 (a+b)+c==(a+c)+b 能否成立？**  
【参考答案】  
两者都不行。在比较float或double时，不能简单地比较。由于计算误差，相等的概率很低。应判断两数之差是否落在区间（-e,e)内。  
这个e应比浮点数的精度大一个数量级。  
**8、全局变量和局部变量有什么区别？是怎么实现的？操作系统和编译器是怎么知道的？**  
【参考答案】  
生命周期不同：全局变量随主程序创建和创建，随主程序销毁而销毁；  
局部变量在局部函数内部，甚至局部循环体等内部存在，退出就不存在； 内存中分配在全局数据区。   
操作系统和编译器通过内存分配的位置来知道的，全局变量分配在全局数据段并且在程序开始运行的时候被加载。局部变量则分配在堆栈里面 。  
**9、Heap(堆) 与 Stack(栈) 的差别**  
【参考答案】  
Heap是堆，stack是栈。 Stack的空间由操作系统自动分配/释放，Heap上的空间手动分配/释放。  
Stack空间有限，Heap是很大的自由存储区 C中的malloc函数分配的内存空间即在堆上,C++中对应的是new操作符。  
程序在编译期对变量和函数分配内存都在栈上进行,且程序运行过程中函数调用时参数的传递也在栈上进行  
为什么需要使用堆，使用堆空间的原因？  
【参考答案】直到运行时才知道一个对象需要多少内存空间；不知道对象的生存期到底有多长。  
**10、const关键字？有哪些作用？**  
【参考答案】  
const关键字至少有下列n个作用：  
（1）欲阻止一个变量被改变，可以使用const关键字。在定义该const变量时，通常需要对它进行初始化，因为以后就没有机会再去改变它了；  
（2）对指针来说，可以指定指针本身为const，也可以指定指针所指的数据为const，或二者同时指定为const；  
（3）在一个函数声明中，const可以修饰形参，表明它是一个输入参数，在函数内部不能改变其值；  
（4）对于类的成员函数，若指定其为const类型，则表明其是一个常函数，不能修改类的成员变量；  
（5）对于类的成员函数，有时候必须指定其返回值为const类型，以使得其返回值不为“左值”。  
**11、谈谈你是怎么认识 C++ 中的模板的？**  
【参考答案】  
模板使程序员能够快速建立具有类型安全的类库集合和函数集合，它的实现，方便了大规模的软件开发。（结合stl更好）  
【参考答案】  
(1)可用来创建动态增长和减小的数据结构  (2)它是类型无关的，因此具有很高的可复用性。  
(3)它在编译时而不是运行时检查数据类型，保证了类型安全（4）它是平台无关的，可移植性（5）可用于基本数据类型  
**12、在 C++的一个类中声明一个 static 成员变量有没有用？**  
【参考答案】  
在C++类的成员变量被声明为 static（称为静态成员变量），意味着它为该类的所有实例所共享，也就是说当某个类的实例修改了该静态成员变量，  
也就是说不管创建多少对象，static修饰的变量只占有一块内存。其修改值为该类的其它所有实例所见；而类的静态成员函数也只能访问静态成员（变量或函数）。static是加了访问控制的全局变量，不被继承。  
**13、基类的析构函数不是虚函数，会带来什么问题？**  
【参考答案】  
派生类的析构函数用不上，会造成资源的泄漏。  
**14、一个栈的入栈序列是 A，B，C，D，E，则栈的不可能的输出序列是？**  
A、EDCBA；B、DECBA；    C、DCEAB；D、ABCDE   
【标准答案】C  
A：a,b,c,d,e进，之后依次出栈；B：a,b,c,d,进，d出，e进，e,c,b,a出  
D：a进a出，b进b出……e进e出；C：的话dce都好办，之后的ab做不到

**C/C++常见笔试题1**1、请填写bool , float,  指针变量 与“零值”比较的if    
语句。  
提示：这里“零值”可以是0, 0.0 , FALSE  或者“空指针”  
。例如int 变量n  与“零值”比较的if  语句为：  
if ( n == 0 )   
if ( n != 0 )   
以此类推。  
（1）请写出bool flag  与“零值”比较的if  语句：  
**【标准答案】if ( flag )    if ( !flag )**  
100 条经典 C语言笔试题目  
（2）请写出float  x   与“零值”比较的if  语句：  
**【标准答案】 const float EPSINON = 0.00001;**  
**if ((x >= - EPSINON) && (x <= EPSINON)**               
不可将浮点变量用“==” 或“！=” 与数字比较，应该设法  
转化成“>=” 或“<=” 此类形式。  
100 条经典 C语言笔试题目  
（3）请写出char  \*p   与“零值”比较的if  语句  
**【标准答案】 if (p == NULL)    if (p != NULL)**  
  
2、以下为Linux下的32 位C 程序，请计算sizeof 的  
值。  
char  str[] = “Hello” ;                                 
char   \*p = str ;                                        
i nt    n = 10;                                                      
请计算  
（1）sizeof (str ) =                  （2）s i zeof ( p ) =                   
（3）sizeof ( n ) =  
【标准答案】  
   
**（1）6、（2）4 、（3 ）4**  
（4）void Func ( char str[100])                             
{                                            
…… ;                                                              
}                                            
请计算sizeof( str ) =                                         
（5）void \* p = malloc( 100 );                                                  
请计算sizeof ( p ) =   
**【标准答案】（4）4、（5）4**  
   
**4、用变量a 给出下面的定义**  
e) 一个有10个指针的数组，该指针是指向一个整型数  
的;  
f)  一个指向有10个整型数数组的指针;  
g) 一个指向函数的指针，该函数有一个整型参数并返  
回一个整型数;  
h) 一个有10个指针的数组，该指针指向一个函数，该  
函数有一个整型参数并返回一个整型数;  
【标准答案】e)int \*  a[10];      f)int (\*a)[10]                                   
g)int (\*a)(int);   h) int (\*a[10])(int)  
**5、设有以下说明和定义：**  
typedef u nion {long i; int k[5]; char c;} DATE;  
struct data { int cat; DATE cow; double dog;} too;  
DATE max;  
则语句printf("%d",sizeof(struct date)+sizeof(max)); 的  
执行结果是：\_\_\_\_\_  
【标准答案】DATE是一个union, 变量公用空间.  里面最  
大的变量类型是int[5],  占用20个字节.  所以它的大小是  
20  
data 是一个struct,  每个变量分开占用空间.  依次为int4 +   
DATE20 + double8 = 32.  
所以结果是20 + 32 = 52.  
当然... 在某些16位编辑器下, int 可能是2字节,那么结果  
是int2 + DATE10 + double8 = 20  
**6、请问以下代码有什么问题：**  
int main()  
{  
char a;  
char \*str=&a;  
strcpy(str,"hello");  
printf(str);  
return 0;  
}  
【标准答案】**没有为str分配内存空间，将会发生异常**  
**问题出在将一个字符串复制进一个字符变量指针所指**  
**地址**。虽然可以正确输出结果，但因为越界进行内在  
读写而导致程序崩溃。  
   
7、请问以下代码有什么问题：  
char\* s="AAA";  
printf("%s",s);  
s[0]='B';  
printf("%s",s);  
有什么错？  
【标准答案】**"AAA" 是字符串常量**。s是指针，指向这个  
字符串常量，所以声明s的时候就有问题。  
cosnt char\* s="AAA";  
然后又因为是常量，所以**对是s[0] 的赋值操作是不合法**  
的。  
8、int (\*s[10])(int)  表示的是什么啊  
【标准答案】**int (\*s[10])(int)  函数指针数组**，每个指针  
指向一个int func(intp aram) 的函数。  
9、c和c++ 中的struct有什么不同？  
【标准答案】c和c++ 中struct的主要区别是**c中的struct  
不可以含有成员函数，而c++ 中的struct可以**。  
**c++ 中struct和class的主要区别在于默认的存取权限不同**，  
struct默认为public ，而class默认为private  
10、void getmemory(char \*p)  
{  
p=(char \*) malloc(100);  
strcpy(p,“hello world”);  
}  
int main( )  
{  
char \*str=NULL;  
getmemory(str);  
printf(“%s/n”,str);  
free(str);  
return 0;  
} 会出现什么问题？  
【标准答案】程序崩溃，**getmemory中的malloc 不能返回  
动态内存**，free （）对str操作很危险。  
   
   
11、char szstr[10];  
strcpy(szstr,"0123456789");  
产生什么结果？为什么？  
【标准答案】**长度不一样，出现段错误**。  
100 条经典 C语言笔试题目  
12、数组和链表的区别？  
【标准答案】数组：**数据顺序存储，固定大小**；  
**链表：数据可以随机存储，大小可动态改变**  
13、void main()  
{  
char aa[10];  
printf(“%d”,strlen(aa));  
}                                            
会出现什么问题？打印结果是是多少？  
【标准答案】**sizeof()和初不初始化，没有关系**，  
**strlen()和初始化有关，打印结果值未知**。  
14、给定结构struct A  
{  
char t:4;  
char k:4;  
unsigned short i:8;  
unsigned long m;  
}; 问sizeof(A) = ?  
【标准答案】8  
15、struct name1{  
char str;  
short x;  
int num;  
} ；求sizeof(name1)?  
【标准答案】8  
16、struct name2{  
char str;  
int num;  
short x;  
}; 求sizeof(name2)？  
【标准答案】12  
17、程序哪里有错误  
wap( int\* p1,int\* p2 )  
{  
int \* p;  //(int)malloc(4); is ok  
\*p = \*p1;  
\*p1 = \*p2;  
\*p2 = \*p;  
}  
【标准答案】**p 为野指针（指向一个已删除的对象或未申请访问受限内存区域的指针）**  
19、(void \*)ptr 和(\*(void\*\*))ptr 的结果是否相同？其  
中ptr为同一个指针。  
【标准答案】**(void \*)ptr 和(\*(void\*\*))ptr 值是相同的**  
20、要对绝对地址0x100000赋值，我们可以用  
(unsigned int\*)0x100000 = 1234;  
那么要是想让程序跳转到绝对地址是0x100000去执行  
，应该怎么做？  
【标准答案】\*((void (\*)( ))0x100000 ) ( );  
首先要**将0x100000强制转换成函数指针**,即:  
(void (\*)())0x100000  
**然后再调用它:**  
\*((void (\*)())0x100000)();  
22、关于内存的思考题（1）你能看出有什么问题？  
<https://blog.csdn.net/qq_17242957/article/details/50628309>  
**void GetMemory(char \*p)   
{   
p = (char \*)malloc(100);   
}   
void Test(void)   
{   
char \*str = NULL;   
GetMemory(str);   
strcpy(str, "hello world");   
printf(str);   
}   
请问运行Test 函数会有什么样的结果？   
答：程序崩溃。   
因为GetMemory 并不能传递动态内存，   
Test 函数中的 str 一直都是 NULL。   
strcpy(str, "hello world");将使程序崩 溃。---因为str没有空间**  
23、关于内存的思考题（2）你能看出有什么问题？  
**char \*GetMemory(void)   
{   
char p[] = "hello world";  //char \*p="hello world"可以，p是局部变量，但是\*p不是常量区的数据。  
return p;   
}   
void Test(void)   
{   
char \*str = NULL;   
str = GetMemory();   
printf(str);   
}**  
   
**字符串是不可直接返回的，因此没用所谓的返回"hello world"之类的说法，你只能返回字符串的首地址。  
当你用char p[]方式定义时，系统在堆栈上创建一个临时数组，然后把hello world内容拷贝进去，因此当你返回p时，实际是返回那个临时数组的首地址。  
改成指针以后，p实际指向的是一个常量字符串"hello world"，而这个字符串是在常量区永远存在的，不是临时变量。可以这么说，p是局部变量，但是\*p不是**  
   
**请问运行Test 函数会有什么样的结果？**  
**答：可能是乱码。**  
**因为GetMemory 返回的是指向“栈内存”**  
**的指针，该指针的地址不是 NULL，但其原**  
**现的内容已经被清除，新内容不可知。**  
25、关于内存的思考题（3）你能看出有什么问题？  
**void GetMemory2(char \*\*p, int num)**  
**{**  
**\*p = (char \*)malloc(num);**  
**printf("\*p=%p\n", \*p);//------------------地址1**  
**}**  
**void Test(void)**  
**{**  
**char \*str = NULL;**  
**GetMemory(&str, 100);**  
**printf("str=%p\n", str); //------------------地址2 == 地址1**  
**strcpy(str, "hello");**  
**printf(str);**  
**}**  
**请问运行Test 函数会有什么样的结果？**  
**答：**  
**（1）能够输出hello**  
**（2）内存泄漏       //没有释放**  
26、关于内存的思考题（4）你能看出有什么问题？  
**void Test(void)**  
**{**  
**char \*str = (char \*) malloc(100);**  
**strcpy(str, “hello”);**  
**free(str);**  
**if(str != NULL)**  
**{**  
**strcpy(str, “world”);**  
**printf(str);**  
**}**  
**}**  
**请问运行Test 函数会有什么样的结果？**  
**答：篡改动态内存区的内容，后果难以预**  
**料，非常危险。**  
**因为free(str);之后，str 成为野指针， （free(str);之后要加上 str=NULL）**  
**if(str != NULL)语句不起作用**  
   
1.为什么指针变量定义时一定要初始化？   
答:因为你首先要理解一点.内存空间不是你分配了才可以使用  
只是你分配了之后使用才安全,为什么要进行对他初始化呢  
因为,如果你没对他初始化,而引用这个指针并却其指向的内存进行修改  
因为指针未被初始化,所以指针所指向的也是随机的,他是个野指针,如果你引用指针,并修改这个指针所指向的内容,而如果这个指针所指向的内容恰好是另外一个程序的数据的话,你将其进行修改了,就会导致另外一个程序可能不能正常运行了.所以使用前一定要进行初始化  
2.指针变量初始化为NULL是什么意思？  
答:意思是说,强指针变量置空,初始化为NULL,使它不指向任何内容,这样引用她也不会出现上面的问题  
   
总之一点,记住在使用指针之前要对它进行初始化操作就可以了  
27、关键字volatile有什么含意? 并给出三个不同的例  
子。  
【参考答案】一个定义为volatile的变量是说这变量可  
能会被意想不到地改变，这样，编译器就不会去假设  
这个变量的值了。精确地说就是，优化器在用到这个  
变量时必须**每次都小心地重新读取这个变量的值，而  
不是使用保存在寄存器里的备份**。下面是volatile变量  
的几个例子：  
1). 并行设备的硬件寄存器（如：状态寄存器）  
2). 一个中断服务子程序中会访问到的非自动变量  
(Non-automatic variables)  
3). 多线程应用中被几个任务共享的变量  
  
28、嵌入式系统经常具有要求程序员去访问某特定的  
内存位置的特点。在某工程中，要求**设置一绝对地址  
为0x67a9的整型变量的值为0xaa66**。编译器是一个纯  
粹的ANSI编译器。写代码去完成这一任务。  
【参考答案】这一问题测试你是否知道为了访问一绝对  
地址**把一个整型数强制转换（typecast ）为一指针是合  
法的**。这一问题的实现方式随着个人风格不同而不同  
。典型的类似代码如下：  
int \* ptr;  
ptr=  (int \*)0x67a9;  
\*ptr = 0xaa55;  
  
29、头文件中的ifndef/define/endif 干什么用？  
【标准答案】**防止该头文件被重复引用**。  
  
30、#include  <filename.h>    和#include    
“filename.h” 有什么区别？  
【标准答案】对于#include  <filename.h>   ，编译器从  
**标准库路径开始搜索**filename.h ;                                  
对于#include  “filename.h” ，编译器从**用户的工作路**  
**径开始搜索**filename.h 。  
   
31、const   有什么用途？（请至少说明两种）  
【标准答案】： （1）可以定义const   **常量**  
（2）const  可以**修饰函数的参数、返回值，甚至函数**  
**的定义体**。  
**被const  修饰的东西都受到强制保护**，可  
以预防意外的变动，能提高程序的健壮性。  
   
32、static有什么用途？（请至少说明两种）  
【标准答案】  
**1. 限制变量的作用域（static全局变量）；  
2. 设置变量的存储域（static局部变量）。**  
  
33、**堆栈溢出一般是**由什么原因导致的？  
【标准答案】**没有回收垃圾资源**。  
  
34、如何引用一个已经定义过的全局变量？  
【标准答案】可以用引用头文件的方式，也可以用  
extern 关键字，如果用**引用头文件方式**来引用某个在  
头文件中声明的全局变理，假定你将那个变量写错了  
，那么在**编译期间会报错**，如果你用**extern 方式**引用  
时，假定你犯了同样的错误，那么在编译期间不会报  
错，而在**连接期间报错**。  
  
35、全局变量可不可以定义在可被多个.C 文件包含的  
头文件中？为什么？  
【标准答案】可以，在不同的C 文件中以static形式来声  
明同名全局变量。可以在不同的C文件中声明同名的全  
局变量，前提是其中只能有一个C文件中对此变量赋初  
值，此时连接不会出错。  
<https://blog.csdn.net/basstal/article/details/52275191>  
从技术上来说，声明不会有链接属性，因为声明不会在可执行映像中分配存储空间；因此，不存在链接器是否容许交叉引用那些存储空间的问题。  
当把（所谓的）全局变量global定义为static时，由于static使定义的变量称为内部链接，所以在各个.c文件中，存在多个同名global但不同等的定义，每个翻译单元中的global维持自己的内存区域，此时链接器不会报告“符号被多重定义”错误。  
   
此时，（所谓的）全局变量并没有达到一般意义上全局变量的效果，相当于每个翻译单元的局部变量。  
36、队列和栈有什么区别？  
【标准答案】**队列先进先出，栈后进先出**。  
37、**Heap与stack的差别。**  
【标准答案】Heap是堆，stack是栈。  
Stack的空间由操作系统自动分配/释放，Heap上的空  
间手动分配/释放。  
Stack空间有限，Heap是很大的自由存储区  
C 中的malloc 函数分配的内存空间即在堆上,C++中对  
应的是new 操作符。  
程序在编译期对变量和函数分配内存都在栈上进行,且  
程序运行过程中函数调用时参数的传递也在栈上进行。  
38、**用宏定义写出swap（x，y），即交换两数。**  
【标准答案】  
#define swap(x, y) (x)=(x)+(y);(y)=(x)–(y);(x)=(x)–(y);  
39、写一个“标准”宏，这个宏输入两个参数并返回较  
小的一个。  
【标准答案】**#define Min(X, Y) ((X)>(Y)?(Y):(X))// 结尾没有 ;**  
40、带参宏与带参函数的区别(至少说出5点)？  
【标准答案】  
                     带参宏                带参函数  
处理时间             编译时                运行时  
参数类型                 无               需定义  
程序长度                变长               不变  
占用存储空间           否                    是  
运行时间           不占运行时间      调用和返回时占  
   
[实参](http://baike.baidu.com/item/%E5%AE%9E%E5%8F%82)如果是[表达式](http://baike.baidu.com/item/%E8%A1%A8%E8%BE%BE%E5%BC%8F)容易出问题  
#define S(r) r\*r  
area=S(a+b);第一步换为area=r\*r;,第二步被换为area=a+b\*a+b;  
正确的宏定义是[#define](http://baike.baidu.com/item/%23define) S(r) ((r)\*(r))  
（2）宏名和参数的括号间不能有空格  
（3）宏替换只作替换，不做计算，不做表达式求解  
（4）函数调用在编译后程序运行时进行，并且分配内存。宏替换在编译前进行，不分配内存  
（5）宏的哑实结合不存在类型，也没有类型[转换](http://baike.baidu.com/item/%E8%BD%AC%E6%8D%A2)。  
（6）[宏](http://baike.baidu.com/item/%E5%AE%8F/2648286)展开使[源程序](http://baike.baidu.com/item/%E6%BA%90%E7%A8%8B%E5%BA%8F)变长，[函数调用](http://baike.baidu.com/item/%E5%87%BD%E6%95%B0%E8%B0%83%E7%94%A8)不会  
（7）宏展开不占运行时间，只占[编译时间](http://baike.baidu.com/item/%E7%BC%96%E8%AF%91%E6%97%B6%E9%97%B4)，函数调用占运行时间（分配内存、保留现场、[值传递](http://baike.baidu.com/item/%E5%80%BC%E4%BC%A0%E9%80%92)、[返回值](http://baike.baidu.com/item/%E8%BF%94%E5%9B%9E%E5%80%BC)。  
   
41、关键字**volatile**有什么含意？  
【标准答案】提示编译器对象的值可能在编译器未监测  
到的情况下改变。  
42、int main()  
{  
int x=3;  
printf("%d",x);  
return 1;  
}  
问函数既然不会被其它函数调用，为什么要返回1？  
【标准答案】mian中，c标准认为0表示成功，非0表示  
错误。具体的值是某中具体出错信息。  
<https://blog.csdn.net/menoy/article/details/54141585>  
<https://blog.csdn.net/pangyemeng/article/details/78853671>  
   
43、已知一个数组tabl e ，用一个宏定义，求出数据的  
元素个数。  
【标准答案】  
#define NTBL(table) (sizeof(table)/sizeof(table[0]))  
   
44、A.c 和B.c两个c文件中使用了两个相同名字的  
static变量,编译的时候会不会有问题?这两个static变量  
会保存到哪里（栈还是堆或者其他的）?  
【标准答案】static的全局变量，表明这个变量仅在本  
模块中有意义，不会影响其他模块。  
他们都放在静态数据区，但是编译器对他们的命名是  
不同的。  
如果要使变量在其他模块也有意义的话，需要使用  
extern 关键字。  
<https://www.cnblogs.com/SarielTang/p/4501456.html>    
[**关于Static、全局变量、局部变量、Extern之间的种种**](https://www.cnblogs.com/SarielTang/p/4501456.html)  
   
45、static全局变量与普通的全局变量有什么区别？  
【标准答案】 static全局变量只初使化一次，防止在其  
他文件单元中被引用;  
全局变量(外部变量)的说明之前再冠以static 就构成了静态的全局变量。  
　　**全局变量本身就是静态存储方式， 静态全局变量当然也是静态存储方式。** 这两者在存储方式上并无不同。  
　　这两者的区别在于非静态全局变量的作用域是整个源程序， 当一个源程序由多个源文件组成时，非静态的全局变量在各个源文件中都是有效的。 而静态全局变量则限制了其作用域， 即只在定义该变量的源文件内有效， 在同一源程序的其它源文件中不能使用它。由于静态全局变量的作用域局限于一个源文件内，只能为该源文件内的函数公用，因此可以避免在其它源文件中引起错误。   
static全局变量只初使化一次，防止在其他文件单元中被引用; 　   
46、static局部变量和普通局部变量有什么区别  
【标准答案】static局部变量只被初始化一次，下一次  
依据上一次结果值；  
　把局部变量改变为静态变量后是改变了它的存储方式即改变了它的**生存期**。把全局变量改变为静态变量后是改变了它的**作用域**，限制了它的使用范围。    
　　static局部变量只被初始化一次，下一次依据上一次结果值  
47、static函数与普通函数有什么区别？  
【标准答案】**static函数在内存中只有一份，普通函数在  
每个被调用中维持一份拷贝**  
static函数与普通函数作用域不同,仅在本文件。只在当前源文件中使用的函数应该说明为内部函数(static修饰的函数)，内部函数应该在当前源文件中说明和定义。对于可在当前源文件以外使用的函数，应该在一个头文件中说明，要使用这些函数的源文件要包含这个头文件.  
static函数在内存中只有一份，普通函数在每个被调用中维持一份拷贝  
   
关于45-47 的参考文章  
   
在C语言中，static的字面意思很容易把我们导入歧途，其实它的作用有三条。  
（1）先来介绍它的第一条也是最重要的一条：**隐藏**。  
当我们同时编译多个文件时，所有未加static前缀的全局变量和函数都具有全局可见性。为理解这句话，我举例来说明。我们要同时编译两个源文件，一个是a.c，另一个是main.c。  
下面是a.c的内容  
char a = 'A'; // global variable  
void msg()   
{  
    printf("Hello\n");   
}  
 下面是main.c的内容  
int main(void)  
{      
    extern char a;    // extern variable must be declared before use  
    printf("%c ", a);  
    (void)msg();  
    return 0;  
}  
程序的运行结果是：  
A Hello  
你可能会问：为什么在a.c中定义的全局变量a和函数msg能在main.c中使用？前面说过，所有未加static前缀的全局变量和函数都具有全局可见性，其它的源文件也能访问。此例中，a是全局变量，msg是函数，并且都没有加static前缀，因此对于另外的源文件main.c是可见的。  
如果加了static，就会对其它源文件隐藏。例如在a和msg的定义前加上static，main.c就看不到它们了。利用这一特性可以在不同的文件中定义同名函数和同名变量，而不必担心命名冲突。Static可以用作函数和变量的前缀，对于函数来讲，static的作用仅限于隐藏，而对于变量，static还有下面两个作用。  
（2）static的第二个作用是保持变量内容的持久。存储在静态数据区的变量会在程序刚开始运行时就完成初始化，也是唯一的一次初始化。共有两种变量存储在静态存储区：全局变量和static变量，只不过和全局变量比起来，static可以控制变量的可见范围，说到底static还是用来隐藏的。虽然这种用法不常见，但我还是举一个例子。  
#include <stdio.h>  
int fun(void){  
    static int count = 10;    // 事实上此赋值语句从来没有执行过  
    return count--;  
}  
int count = 1;  
int main(void)  
{      
    printf("global\t\tlocal static\n");  
    for(; count <= 10; ++count)  
        printf("%d\t\t%d\n", count, fun());      
      
    return 0;  
}  
 程序的运行结果是：  
global          local static  
1               10  
2               9  
3               8  
4               7  
5               6  
6               5  
7               4  
8               3  
9               2  
10              1  
（3）static的第三个作用是默认初始化为0。其实全局变量也具备这一属性，因为全局变量也存储在静态数据区。在静态数据区，内存中所有的字节默认值都是0x00，某些时候这一特点可以减少程序员的工作量。比如初始化一个稀疏矩阵，我们可以一个一个地把所有元素都置0，然后把不是0的几个元素赋值。如果定义成静态的，就省去了一开始置0的操作。再比如要把一个字符数组当字符串来用，但又觉得每次在字符数组末尾加’\0’太麻烦。如果把字符串定义成静态的，就省去了这个麻烦，因为那里本来就是’\0’。不妨做个小实验验证一下。  
#include <stdio.h>  
int a;  
int main(void)  
{  
    int i;  
    static char str[10];  
    printf("integer: %d;  string: (begin)%s(end)", a, str);  
    return 0;  
}  
程序的运行结果如下  
integer: 0; string: (begin)(end)  
最后对static的三条作用做一句话总结。首先static的最主要功能是隐藏，其次因为static变量存放在静态存储区，所以它具备持久性和默认值0。  
以上内容出自博客园Mr. Write之手，写的相当清晰易懂，存档方便复习。原文地址：<http://www.cnblogs.com/dc10101/archive/2007/08/22/865556.html>  
全局变量(外部变量)的说明之前再冠以static 就构成了静态的全局变量  
。全局变量本身就是静态存储方式，静态全局变量当然也是静态存储方  
式。这两者在存储方式上并无不同。这两者的区别虽在于非静态全局变  
量的作用域是整个源程序，当一个源程序由多个源文件组成时，非静态  
的全局变量在各个源文件中都是有效的。而静态全局变量则限制了其作  
用域，即只在定义该变量的源文件内有效，在同一源程序的其它源文  
件中不能使用它。由于静态全局变量的作用域局限于一个源文件内，只  
能为该源文件内的函数公用，因此可以避免在其它源文件中引起错误。  
从以上分析可以看出，把局部变量改变为静态变量后是改变了它的存储  
方式即改变了它的生存期。把全局变量改变为静态变量后是改变了它的  
作用域，限制了它的使用范围。  
static函数与普通函数作用域不同。仅在本文件。只在当前源文件中使  
用的函数应该说明为内部函数(static)，内部函数应该在当前源文件中  
说明和定义。对于可在当前源文件以外使用的函数，应该在一个头文件  
中说明，要使用这些函数的源文件要包含这个头文件。  
   
Extern与Static：  
    首先，我要告诉你static与extern是一对“水火不容”的家伙，也就是说extern和static不能同时修饰一个变量；其次，static修 饰的全局变量声明与定义同时进行，也就是说当你在头文件中使用static声明了全局变量后，它也同时被定义了；最后，static修饰全局变量的作用域 只能是本身的编译单元，也就是说它的“全局”只对本编译单元有效，其他编译单元则看不到它,如:  
    test1.h:  
    #ifndef TEST1H  
    #define TEST1H  
    static char g\_str[] = "123456";   
    void fun1();  
    #endif  
    test1.cpp:  
    #include "test1.h"  
      
    void fun1()  
    {  
        cout << g\_str << endl;  
    }  
      
    test2.cpp  
    #include "test1.h"  
      
    void fun2()  
    {  
        cout << g\_str << endl;  
    }  
      
    以上两个编译单元可以连接成功, 当你打开test1.obj时，你可以在它里面找到字符串"123456", 同时你也可以在test2.obj中找到它们，它们之所以可以连接成功而没有报重复定义的错误是因为虽然它们有相同的内容，但是存储的物理地址并不一样， 就像是两个不同变量赋了相同的值一样，而这两个变量分别作用于它们各自的编译单元。  
    也许你比较较真，自己偷偷的跟踪调试上面的代码,结果你发现两个编译单元（test1, test2）的g\_str的内存地址相同，于是你下结论static修饰的变量也可以作用于其他模块，但是我要告诉你，那是你的编译器在欺骗你，大多数编 译器都对代码都有优化功能，以达到生成的目标程序更节省内存，执行效率更高，当编译器在连接各个编译单元的时候，它会把相同内容的内存只拷贝一份，比如上 面的"123456", 位于两个编译单元中的变量都是同样的内容，那么在连接的时候它在内存中就只会存在一份了， 如果你把上面的代码改成下面的样子，你马上就可以拆穿编译器的谎言:  
    test1.cpp:  
    #include "test1.h"  
      
    void fun1()  
    {  
        g\_str[0] = 'a';  
        cout << g\_str << endl;  
    }  
    test2.cpp  
    #include "test1.h"  
      
    void fun2()  
    {  
        cout << g\_str << endl;  
    }  
      
    void main()  
    {  
        fun1(); // a23456  
        fun2(); // 123456  
    }  
      
    这个时候你在跟踪代码时，就会发现两个编译单元中的g\_str地址并不相同，因为你在一处修改了它，所以编译器被强行的恢复内存的原貌，在内存中存在了两份拷贝给两个模块中的变量使用。  
    正是因为static有以上的特性，所以一般定义static全局变量时，都把它放在原文件中而不是头文件，这样就不会给其他模块造成不必要的信息污染，同样记住这个原则吧！  
   
   
48、程序的局部变量存在于\_\_\_ 中，全局变量存在于  
\_\_\_\_中，动态申请数据存在于\_\_\_ 中。  
【标准答案】程序的局部变量存在于**栈**(stack) 中，全局  
变量存在于**静态数据区**中，动态申请数据存在于**堆**（  
heap）中。  
   
49、什么是预编译，何时需要预编译：  
【标准答案】１、总是使用不经常改动的大型代码体  
。  
２、程序由多个模块组成，所有模块都使用一组标准  
的包含文件和相同的编译选项。在这种情况下，可以  
将所有包含文件预编译为一个预编译头。  
<https://blog.csdn.net/u011146511/article/details/69568830>  C语言中的三大预编译功能  
这三种[预处理](http://baike.baidu.com/item/%E9%A2%84%E5%A4%84%E7%90%86)包括：宏定义、[文件包含](http://baike.baidu.com/item/%E6%96%87%E4%BB%B6%E5%8C%85%E5%90%AB)、[条件编译](http://baike.baidu.com/item/%E6%9D%A1%E4%BB%B6%E7%BC%96%E8%AF%91)。  
   
50、用**两个栈实现一个队列**的功能？要求给出算法和  
思路！  
【参考答案】设2个栈为A,B, 一开始均为空.  
入队:  
将新元素push入栈A;  
出队:  
(1)判断栈B 是否为空；  
(2)如果不为空，则将栈A中所有元素依次pop 出并  
push到栈B；  
(3)将栈B 的栈顶元素pop 出；  
   
51、对于一个频繁使用的短小函数,在C 语言中应用什  
么实现,在C++ 中应用什么实现?  
【标准答案】**c用宏定义，c++ 用inline**  
   
52、1. 用预处理指令#define  声明一个常数，用以表  
明1年中有多少秒（忽略闰年问题）  
【参考答案】#define SECONDS\_PER\_YEAR (60 \* 60   
\* 24 \* 365)UL  
   
 #define SECONDS\_PER\_YEAR (60 \* 60 \* 24 \* 365)  
printf("SECONDS\_PER\_YEAR=%ul\n", SECONDS\_PER\_YEAR);  
   
   
53、Typedef 在C 语言中频繁用以声明一个已经存在的  
数据类型的同义字。也可以用预处理器做类似的事。例  
如，思考一下下面的例子：  
**#define dPS struct s \*  
typedef struct s \* tPS;**  
以上两种情况的意图都是要定义dPS 和tPS 作为一个  
指向结构s指针。哪种方法更好呢？（如果有的话）为  
什么？  
【参考答案】这是一个非常微妙的问题，任何人答对这个问题（正当的原  
因）是应当被恭喜的。答案是：typedef 更好。思考下面的例子：  
dPS p 1,p2;  
tPS p3,p4;  
第一个扩展为  
struct s \* p1, p2;  
上面的代码定义p1为一个指向结构的指，p2为一个实际的结构，这也许  
不是你想要的。第二个例子正确地定义了p3 和p4 两个指针。  
  
54、在C++  程序中调用被C 编译器编译后的函数，  
为什么要加extern “C”？  
【标准答案】C++ 语言支持函数重载，C 语言不支持函  
数重载。函数被C++ 编译后在库中的名字与 C 语言的  
不同。假设某个函数的原型为：void foo(int x, int y);   
该函数被C 编译器编译后在库中的名字为\_foo ，而  
C++ 编译器则会产生像\_foo\_int\_int之类的名字。 C++  
提供了C 连接交换指定符号extern“C”来解决名字匹配  
问题。  
   
55、请简述以下两个for  循环的优缺点。  
for (i=0; i<N; i++)  
{  
 if (condition)  
 DoSomething();  
 else  
 DoOtherthing();  
}  
  
if (condition)  
{  
 for (i=0; i<N; i++)  
 DoSomething();  
}  
else  
{  
 for (i=0; i<N; i++)  
 DoOtherthing();  
}  
   
【标准答案】  
优点：程序简洁  
缺点：多执行了N-1次逻辑判断，并且打断了循环“流水线”作业，使得编译器不能对循环进行优化处理，降低了效率。  
优点：循环的效率高  
缺点：程序不简洁  
  
56、语句for(  ；1  ；) 有什么问题？它是什么意思？  
【标准答案】死循环，和while(1)相同。  
57、do……while和while……do有什么区别？  
【标准答案】前一个循环一遍再判断，后一个判断以后  
再循环。  
  
58、请写出下列代码的输出内容  
#include <stdio.h>  
int main()  
{  
int a,b,c,d;  
a=10;  
b=a++; //b=10, a=11  
c=++a; //c=12, a=12  
d=10\*a++;//d=120, a=13  
printf("b，c ，d：%d，%d，%d"，b，c，d ）;  
return 0;  
}  
【标准答案】10，12，120  
  
59、unsigned char \*p1;  
unsigned long \*p2;  
p1=(unsigned char \*)0x801000;  
p2=(unsigned long \*)0x810000;  
请问p1+5= ;  
p2+5= ;  
【标准答案】0x801005、0x810020  
unsigned char \*p1;  
unsigned long \*p2;  
p1=(unsigned char \*)0x801000;  
p2=(unsigned long \*)0x810000;  
printf("p1=%x, p1+5=%x\n", p1, p1+5);  
printf("p2=%x, p2+5=%x\n", p1, p2+5);  
p1=801000, p1+5=801005  
p2=801000, p2+5=810014  
  
60、main()  
{  
int a[5]={1,2,3,4,5};  
int \* ptr=(int\*)(&a+1);  
**printf(“%d，%d”,\*(a+1),\*(ptr-1));**  
}  
请问输出：  
**【标准答案】2,5**  
int a[5]={1,2,3,4,5};  
int \* ptr=(int\*)(&a+1);  //此时a作为数组的首地址，+1的跨度为整个数组的长度，ptr指向数组a后面的空间地址。  
printf(“%d，%d”,\*(a+1),\*(ptr-1)); //a+1代表a[1], \*a[1]指向2,   (ptr-1)代码a数组的最后一个元素的地址。  
a[0], a[1], a[2], a[3], a[4], ptr  
   
printf("&a=%p\n", &a);  
printf("&a[0]=%p\n", &a[0]);  
printf("&a[1]=%p\n", &a[1]);  
printf("&a[2]=%p\n", &a[2]);  
printf("&a[3]=%p\n", &a[3]);  
printf("&a[4]=%p\n", &a[4]);  
printf("ptr=%p\n", ptr);  
&a=0028FF10  
&a[0]=0028FF10  
&a[1]=0028FF14  
&a[2]=0028FF18  
&a[3]=0028FF1C  
&a[4]=0028FF20  
ptr=0028FF24  
   
61、请问下面程序有什么错误?  
 int a[60][250][1000],i,j,k;  
  for(k=0;k<1000;k++)  
   for(j=0;j<250;j++)  
     for(i=0;i<60;i++)  
      a[i][j][k]=0;  
【标准答案】把循环语句内外换一下。  
（编译的时候没错，运行的时候出错，但这个数组太大，如果放在栈中，还是会溢出，要作为全局变量）  
   
<https://www.cnblogs.com/ios8/p/ios-c-test.html>  [iOS求职之C语言面试题](https://www.cnblogs.com/ios8/p/ios-c-test.html)  
   
62、以下是求一个数的平方的程序,请找出错误:  
#define SQUARE(a)((a)\*(a))  
int a=5;  
int b;  
b=SQUARE(a++);  
【标准答案】宏在预编译时会以替换的形式展开，仅仅  
会替换。涉及到宏的地方，不要用++-- ，标准中对此  
没有规定，因此最终结果将会依赖于不同的编译器。  
执行程序的答案可能是25、也有可能是36。  
dev-c++，windows下为b=25，a=7  
   
63、#define Max\_CB 500  
void LmiQueryCSmd(StructMSgCB \* pmsg)  
{  
unsigned char ucCmdNum;  
......    
for(ucCmdNum=0;ucCmdNum<Max\_CB;ucCmdN  
um++)  
{  
......;  
}                                            
} 这段代码执行有什么问题？  
【标准答案】**死循环**  
unsigned char //无符号字符型表示范围0~255  
char // 有符号字符型 表示范围-128~127  
   
64、嵌入式系统中经常要用到无限循环，你怎么用C  
编写死循环。  
【标准答案】while(1){}或者for(;;)  
100 条经典 C语言笔试题目  
65、程序输出结果是？  
【标准答案】8 ，10，12，14，16  
  
66、  
【标准答案】16  
  
67、int modifyvalue()  
{   
return(x+=10);  
}  
int changevalue(int x )  
{  
return(x+=1);  
}  
void m ain()  
{  
int x =10;  
x++; //x=11  
changevalue(x);  
x++; //x=12  
modifyvalue();  
printf("First output:%d\n",x);  
x++;//x=13  
changevalue(x);  
printf("Second output:%dn",x);  
modifyvalue();  
printf("Thirdoutput:%dn",x);  
}输出?  
【标准答案】12、13、13  
  
68、不能做switch()的参数类型是：  
【标准答案】switch 的参数不能为实型。  
得是整型  
  
69、请写出下列代码的输出内容  
＃i nclude<stdio.h>  
main()  
{  
int a,b,c,d;  
a=10;  
b=a++; //b=10, a=11  
c=++a; //c=12, a=12  
d=10\*a++;//d=120, a = 13  
printf("b，c ，d：%d，%d，%d"，b，c，d ）;  
return 0;  
}  
【标准答案】10，12，120  
  
70、找出程序的错误。  
【标准答案】  
  
71、一语句实现x是否为2 的若干次幂的判断。  
【参考答案】 void mai n()                                                  
{                                            
i nt a;                                                           
scanf(“%d”,&a);                                      
printf(“%c”,**(a)&(a-1)?’n’:’y’**); //   若是打印y，否则n            
}  
72、中断是嵌入式系统中重要的组成部分，这导致了很多编译开发商提  
供一种扩展—让标准C 支持中断。具代表事实是，产生了一个新的关键字  
\_\_interrupt 。下面的代码就使用了**\_\_interrupt 关键字去定义了一个中断服  
务子程序(ISR)**，请评论一下这段代码的。  
\_\_interrupt double compute\_area (double radius)  
{  
double area = PI \* radius \* radius;  
printf(" Area = %f", area);  
return area;  
}  
【参考答案】这个函数有太多的错误了，以至让人不知从何说起了：  
1). **ISR 不能返回一个值**。如果你不懂这个，那么你不会被雇用的。  
2). **ISR 不能传递参数**。如果你没有看到这一点，你被雇用的机会等同第  
一项。  
3). 在许多的处理器/编译器中，浮点一般都是不可重入的。有些处理器/编  
译器需要让额处的寄存器入栈，有些处理器/编译器就是不允许在ISR 中做  
浮点运算。此外，**ISR 应该是短而有效率的，在ISR 中做浮点运算是不明  
智的**。  
4). 与第三点一脉相承，**printf() 经常有重入和性能上的问题**。如果你丢掉  
了第三和第四点，我不会太为难你的。不用说，如果你能得到后两点，那  
么你的被雇用前景越来越光明了。  
  
73、下面的代码输出是什么，为什么？  
void foo(void)  
{  
unsigned int a = 6;  
int b = -20;  
(a+b> 6)? puts("> 6") : puts("<= 6");  
}  
【参考答案】这个问题测试你是否懂得C 语言中的整数自动转换原则，  
我发现有些开发者懂得极少这些东西。不管如何，这无符号整型问题的答  
案是输出是“>6” 。原因是当表达式中存在**有符号**类型和**无符号**类型时所有  
的数都**自动转换为无符号类型**。因此-20 变成了一个非常大的正整数，所  
以该表达式计算出的结果大于6 。这一点对于应当频繁用到无符号数据类  
型的嵌入式系统来说是丰常重要的。如果你答错了这个问题，你也就到了  
得不到这份工作的边缘。  
printf("b=%u\n", b);  //b=4294967276  
a+b = 6+(-20) = 6 + 4294967276 = 4294967282  
   
   
74、评价下面的代码片断：  
unsigned int zero = 0;  
unsigned int compzero =  0xFFFF;  
/\*1‘s complement of zero \*/  
【参考答案】对于一个int型不是16位的处理器为说，上面的代码是不正  
确的。应编写如下：  
**unsigned int compzero = ~0;**  
这一问题真正能揭露出应试者是否懂得**处理器字长的重要性**。在我的经  
验里，好的嵌入式程序员非常准确地明白硬件的细节和它的局限，然而  
PC机程序往往把硬件作为一个无法避免的烦恼。  
到了这个阶段，应试者或者完全垂头丧气了或者信心满满志在必得。如  
果显然应试者不是很好，那么这个测试就在这里结束了。但如果显然应  
试者做得不错，那么我就扔出下面的追加问题，这些问题是比较难的，  
我想仅仅非常优秀的应试者能做得不错。提出这些问题，我希望更多看  
到应试者应付问题的方法，而不是答案。不管如何，你就当是这个娱乐  
吧…  
unsigned int zero = 0;  
unsigned int compzero1 = 0xFFFF;  
unsigned int compzero2 = ~0;  
printf("compzero1=%u\n", compzero1);   
printf("compzero2=%u\n", compzero2);   
compzero1=65535  
compzero2=4294967295  
   
   
75、下面的代码片段的输出是什么，为什么？  
char \*ptr;  
if ((ptr = (char \*)malloc(0)) == NULL)  
puts("Gota null pointer");  
else  
puts("Gota valid pointer");  
【参考答案】这个你可以先大胆猜测下，然后再用你的  
编译器尝试着编译下~~  
char cc='a';  
char \*ptr;  
ptr = &cc;  
printf("ptr=%p, \*ptr=%c\n", ptr,\*ptr);  
if ((ptr = (char \*)malloc(0)) == NULL)  
    puts("Gota null pointer");  
else  
    puts("Gota valid pointer");  
      
printf("ptr=%p, \*ptr=%c\n", ptr,\*ptr);  
ptr=0028FF2F, \*ptr=a  
Gota valid pointer  
ptr=00381720, \*ptr=`  
malloc(0)后，ptr地址发生变更  
   
100 条经典 C语言笔试题目  
欢迎进入C语言程序笔试面试，编写程序代码单元。  
76、编写strcpy 函数  
已知strcpy 函数的原型是 char \*strcpy(char \*strDest,   
const char \*strSrc);其中strDest是目的字符串，  
strSrc 是源字符串。  
（1）不调用C++/C 的字符串库函数，请编写函数  
strcpy 。  
（2）strcpy 能把 strSrc 的内容复制到strDest，为什  
么还要char \*  类型的返回值？  
<https://blog.csdn.net/okawari_richi/article/details/57411796> strcpy()函数详解  
char \* strcpy(char \*dst,const char \*src)   
{  
 if((dst==NULL)||(src==NULL))  
   
 return NULL;   
   
 char \*ret = dst; *//[1]*  
   
 while ((\*dst++=\*src++)!='\0'); *//[2]*  
   
 return ret;*//[3]*  
}  
（1）const 修饰：源字符串参数用const修饰，防止修改源字符串；  
（2）空指针检查：源指针和目的指针都有可能会出现空指针的情况，所以应该对其进行检查；  
（3）为什么要设置ret 指针以及返回ret指针的位置[3]，由于目的指针dst已经在进行移动了，所以用辅助指针ret表明首指针；  
（4）以上所示[2]处，为简单的字符串的复制过程，正好表明strcpy函数遇到'\0'将会停止；  
   
   
77、写出二分查找的代码。  
【参考答案】   
int binary\_search(int\* arr, int key, int n)  
 {  
 int low=0;  
 int mid;  
 int high=n-1;  
 while(low<=high)  
 {  
 mid = (low+high)/2;  
 if(key < arr[mid])  
 {  
 high=mid-1;  
 }  
 else if(key>arr[mid])  
 {  
 low=mid+1;  
 }  
 else  
 {  
 return mid;  
 }  
 }  
 return -1;  
 }  
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 int it[] = {0,1,2,3,4,5,6,7};  
 int index = binary\_search(it, 7, 8);  
 printf("index=%d\n", index); *//index=7*  
 }  
   
 return 0;  
}  
  
78、请编写一个C 函数，该函数给出一个字节中被置  
1  的位的个数。  
【参考答案】  
   
unsigned int TestAsOne1(char log)  
{  
 int i;  
 unsigned int num=0, val;  
 for(i=0; i<8; i++)  
 {  
 val = log >> i; *// 移位*  
 val &= 0x01; *// 与1 相与*  
 if(val)  
 num++;  
 }  
 return num;  
}  
  
79、请编写一个C 函数，该函数将给定的一个字符串  
转换成整数。  
【参考答案】  
int Invert(char\* str)  //此实现不能处理负数和非数字的字符串  
{   
int num =0;   
while(\*str!='\0')   
{   
int d igital=\*str-48;   
num=num\*10+digital;   
str=str+1;   
}   
return num;   
}   
int Invert2(char\* str)  
{  
 if(str==NULL)  
 return -1;  
   
 int num=0, negFlag=0;  
 if(str[0] == '-')  
 {  
 negFlag=1;  
 str++;  
 }  
 while(\*str!='\0')  
 {  
 if(\*str>='0' && \*str<='9')  
 {  
 num = num\*10 + (\*str-'0');  
 }  
 else  
 {  
 num=-1;  
 break;  
 }  
 str++;  
 }  
   
 if(negFlag)  
 num \*= -1;  
   
 return num;  
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char \*str1="1234";  
 int str\_i1= Invert2(str1);  
 printf("str1=%s, str\_i1=%d\n", str1, str\_i1);  
 char \*str2="-1234";  
 int str\_i2= Invert2(str2);  
 printf("str2=%s, str\_i2=%d\n", str2, str\_i2);  
 char \*str3="-12s4";  
 int str\_i3= Invert2(str3);  
 printf("str3=%s, str\_i3=%d\n", str3, str\_i3);  
 }  
 return 0;  
}  
*/\**  
*str1=1234, str\_i1=1234*  
*str2=-1234, str\_i2=-1234*  
*str3=-12s4, str\_i3=1*  
*\*/*  
  
80、请编写一个C 函数，该函数将给定的一个整数转  
换成字符串。  
【参考答案】  
void IntToCharChange(int num,  char\* pval)  //此实现，转换负数有问题  
{   
char strval[100];   
int i , j;   
int val0 = 0;   
int val1 = 0;   
val0 = num;   
for(i=0; i<100; i++)   
{   
val1 = val0 % 10; //取余  
val0 = val0 / 10; // 取整  
strval[i] = val1 + 48;  // 数字—字符  
if(val0 < 10)   
{   
i++;   
strval[i] = val0 + 48;   
break;   
}   
}   
for(j=0; j<=i; j++)  // 倒置  
pval[j] = strval[i-j];   
pval[j] = '\0';   
}  
void IntToCharChange1(int num, char\* pval)  
{  
 *//判断正负号*   
 char negFlag=0;  
 if(num<0)  
 {  
 negFlag=1;  
 }  
   
 *//计算字符串长度*   
 int len=0;  
 int tmp\_num=num;  
 if(negFlag)  
 {   
 tmp\_num \*= -1;  
 len++;  
 }   
 while(tmp\_num>0)  
 {  
 len++;  
 tmp\_num /= 10;  
 }   
 cout << "len=" << len << endl;  
   
 *//转成字符串*   
 tmp\_num = num;   
 if(negFlag)  
 {   
 tmp\_num \*= -1;  
 pval[0] = '-';  
 }   
   
 int val;  
 int i=0;  
 while(tmp\_num>0)  
 {  
 val = tmp\_num%10;  
 pval[len-i-1]= val+'0';  
 tmp\_num=tmp\_num/10;   
 i++;  
 }  
 pval[len] = '\0';  
}   
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 int num1 = 1234;  
 char \*str1;  
 IntToCharChange1(num1, str1);  
 cout << str1 << endl;  
   
 int num2 = -1234;  
 char \*str2;  
 IntToCharChange1(num2, str2);  
 cout << str2 << endl;  
 }  
 return 0;  
}  
   
*/\**  
*len=4*  
*1234*  
*len=5*  
*-1234*  
*\*/*  
81、实现strcmp 函数。  
【参考答案】  
int m ystrcmp(const c har\* str1, const char\* str2)  
{  
assert((str1 != NULL) && (str2 != NULL));  
int r et = 0;  
while (!(ret = \*(unsigned char\*)str1 - \* (unsigned char\*)str2) && \*str2)  
{  
str1++;  
str2++;  
}  
if (ret > 0)  
ret = 1;  
else if (ret < 0)  
ret = -1;  
return ret;  
}  
*//str1和str2可以是字符串常量或者字符串变量，返回值为整形。返回结果如下规定：*  
*//①str1小于str2，返回负值或者-1（VC返回-1）；②str1等于str2，返回0；*  
*//③str1大于str2，返回正值或者1（VC返回1）；*  
int mystrcmp(const char\* str1, const char\* str2)  
{  
 assert((str1 != NULL) && (str2 != NULL));  
 int ret=0;  
 while((ret=\*(unsigned char\*)str1-\*(unsigned char\*)str2)==0 && \*str2)  
 {  
 str1++;  
 str2++;  
 }   
   
 if (ret > 0)  
 ret = 1;  
 else if (ret < 0)  
 ret = -1;  
   
 return 0;  
}   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char \*str1="hellod";  
 char \*str2="hello";  
 int res = mystrcmp1(str1, str2);  
 cout << res << endl;  
 }  
 return 0;  
}  
82、请编写一个C 函数，该函数将一个字符串逆序。  
【参考答案】  
void AntitoneValue(cha r\* father, char\* child)   
{   
int i ;   
char source[100];   
int j = 0;   
while(father[j]) //放入source ，[j] 为长度  
{   
source[j] = father[j];   
j++;   
if(j > 99)   
return;   
}   
source[j] = '\0';   
for(i=0; i<j; i++)   
child[i] = source[j-i-1];  // 反序  
child[i] = '\0';   
}   
void AntitoneValue(char\* father, char\* child)  
{  
 if(father == NULL)  
 return;  
   
 int len = strlen(father);  
 int i;  
 for(i=0; i<len; i++)  
 {  
 child[i]=father[len-i-1];  
 }   
 child[len]='\0';  
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char \*str1="dongj1223";  
 char str2[100];  
 AntitoneValue(str1, str2);  
 cout << str2 << endl;  
 }  
 return 0;  
}  
*//3221jgnod*  
83、请编写一个C 函数，该函数在给定的内存区域搜  
索给定的字符，并返回该字符所在位置索引值。  
【参考答案】  
int s earch(char\* cpSource, intn , char ch)  // 起始地址，搜索长度，目标字符  
{  
int i;  
for(i=0; i<n && \*(cpSource+i) != ch; ++i);  
return i;  
}  
int search(char\* cpSource, int n , char ch) *// 起始地址，搜索长度，目标字符*  
{  
 int res;   
 int i;  
 for(i=0; i<n && \*(cpSource+i)!=ch; i++)  
 {   
 ; *//不相等，则继续；相等则退出；超出范围也退出；*  
 }   
   
 if(i==n)  
 res = -1; *//考虑找不到的情况，返回-1*  
 else  
 res = i; *//找到，则返回索引值*  
   
 return res;  
}  
  
84、请编写一个C 函数，该函数在一个字符串中找到  
可能的最长的子字符串，该字符串是由同一字符组成  
的。  
【参考答案】  
int C hildString(char\*p)     // 自己写  
{     
char \*q =p;  
int s tringlen=0, i=0,j=1,len=0,maxlen=1;     
while(\*q!=’\0’)          //不能用strlen, 求得长度stringlen  
{  
Stringlen++;  
q++;  
}  
while( i<  String len )      
{     
if(\*(p+i)==\*(p+j)& & j< St ri ngle n )     
{     
len++;                    // 统计子串长度  
i++;  
j++;     
}     
else     
{     
if(len>maxlen)           // 统计最大子串长度  
{     
maxlen=len+1;     
len=0;  
}    
else  
len=0;  
i++;  
j++;  
}     
}    
retu rn   ma xlen;      
}  
int ChildString(char\*p) *// 自己写*  
{   
 char \*q = p;  
 int strlen = 0;   
 while(\*q != '\0') *//不能用strlen, 求得长度stringlen*  
 {  
 strlen++;  
 q++;  
 }  
   
 int i, sublen=1, maxlen=0;   
 for(i=0; i<strlen; i++)   
 {   
 if(\*(p+i)==\*(p+i+1) && (i+1<strlen))   
 {   
 sublen++; *// 统计子串长度*  
 }   
 else   
 {   
 if(sublen > maxlen) *// 统计最大子串长度*  
 {   
 maxlen = sublen;   
 }   
 sublen = 1;  
 }   
 }   
 return maxlen;   
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char \*str = "hellllo";  
 int len = ChildString(str);  
 cout << len << endl;  
 }  
 return 0;  
}  
   
   
   
85、华为面试题：怎么判断链表中是否有环？  
【参考答案】答：用两个指针来遍历这个单向链表，第  
一个指针p1，每次走一步；第二个指针p2，每次走两  
步；当p2 指针追上p1的时候，就表明链表当中有环  
路了。  
int testLinkRing(Link \*head)  
{  
Link \*t1=head,\*t2=head;  
while( t1->next && t2->next)  
{  
t1 = t1->next;  
if (NULL == (t2 = t2->next->next))  
return 0; // 无环  
if (t1 == t2)  
return 1;  
}  
return 0;  
}  
int testLinkRing1(Link \*head)  
{  
 Link \*t1=head,\*t2=head;  
 while( t1->next && t2->next)  
 {  
 t1 = t1->next;  
 if (NULL == (t2 = t2->next->next))  
 return 0; *// 无环*  
 if (t1 == t2)  
 return 1;  
 }  
 return 0;  
}  
int testLinkRing2(Link \*head)  
{  
 Link \*slow=head, \*fast=head;  
 while(fast->next && fast->next->next)  
 {  
 slow=slow->next;  
 fast=fast->next->next;  
 if(slow==fast) *//相遇则有环*   
 return 1;  
 }  
 return 0;  
}   
   
Link \*getLinkRingNode(Link \*head)  
{  
 Link \*slow=head, \*fast=head;  
 while(fast->next && fast->next->next)  
 {  
 slow=slow->next;  
 fast=fast->next->next;  
 if(slow==fast) *//相遇则有环*   
 {  
 printf("slow's val = %d\n", slow->val);  
 break;  
 }  
 }  
   
 if(fast->next==NULL || fast->next->next==NULL)  
 return NULL;  
   
 slow=head;  
 while(slow != fast)  
 {  
 slow=slow->next;  
 fast=fast->next;  
 }  
   
   
 return slow;  
}   
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 Link \*node1 = (Link\*)malloc(sizeof(Link));  
 node1->val=1;  
 node1->next=NULL;  
   
 Link \*node2 = (Link\*)malloc(sizeof(Link));  
 node2->val=2;  
 node2->next=NULL;  
   
 Link \*node3 = (Link\*)malloc(sizeof(Link));  
 node3->val=3;  
 node3->next=NULL;  
   
 Link \*node4 = (Link\*)malloc(sizeof(Link));  
 node4->val=4;  
 node4->next=NULL;  
   
 node1->next=node2;  
 node2->next=node3;  
 node3->next=node4;  
 node4->next=node2;  
   
 int res1 = testLinkRing1(node1);   
 int res2 = testLinkRing2(node1);   
 printf("testLinkRing1=%d, testLinkRing2=%d\n", res1, res2);   
   
 Link \*ring\_node = getLinkRingNode(node1);   
 if(ring\_node)   
 printf("ring\_node->val=%d\n", ring\_node->val);   
 else  
 printf("ring\_node is null\n");   
   
 }  
 return 0;  
}  
   
*/\**  
*testLinkRing1=1, testLinkRing2=1*  
*slow's val = 4*  
*ring\_node->val=2*  
*\*/*  
   
   
   
86、有一浮点型数组A, 用C 语言写一函数实现对浮点  
数组A进行降序排序,并输出结果,要求要以数组A 作为  
函数的入口.( 建议用冒泡排序法)  
【参考答案】  
void BubbleSort(double a rr[], int n)   
{   
int i，j ；  
int e xchange = 1 ；// 交换标志  
for(i=1;i<n;i++)  
{ // 最多做n-1趟排序  
exchange=0 ；// 本趟排序开始前，交换标志应为假  
for(j=n-1;j>=i ；j--) //对当前无序区R[i..n]自下向上扫描  
if(arr[j+1] > arr [ j])  
{// 交换记录  
arr[0]=arr[j+1]；//R[0]不是哨兵，仅做暂存单元  
arr[j+1]=arr[j]；  
arr[j]=arr[0]；  
exchange=1 ；// 发生了交换，故将交换标志置为真  
}   
if(!exchange) // 本趟排序未发生交换，提前终止算法  
return ；  
} //endfor(外循环)   
}  
void BubbleSort(double arr[], int n)  
{  
 int i,j;  
 int swapFlag=0;  
 for(i=0; i<n-1; i++)  
 {  
 swapFlag=0;  
 for(j=0; j<n-i-1; j++)  
 {  
 if(arr[j]< arr[j+1])  
 {  
 double temp=arr[j];  
 arr[j]=arr[j+1];  
 arr[j+1]=temp;  
 swapFlag=1;  
 }  
 }  
 if(swapFlag==0)  
 break;  
 }  
}   
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 double arr[5]={3.1,2.3,5.6,4.1,1.2};  
 BubbleSort(arr, 5);  
 int i=0;  
 for(i=0; i<5; i++)  
 {  
 cout << arr[i] << " ";  
 }  
 cout << endl;   
 }  
 return 0;  
}  
   
   
void merge(double arr[], int start, int center, int end, int n)  
{  
 int mid=center+1;  
 double tmp\_arr[n];  
 printf("start=%d, end=%d, length=%d\n", start, end, n);  
 *//double \*tmp\_arr=(double\*)malloc(5 \* sizeof(double));*   
 int start\_backup=start;  
 int tmp\_index=start;  
 while(start<=center && mid<=end)  
 {  
 if(arr[start] >= arr[mid])  
 {  
 tmp\_arr[tmp\_index++] = arr[start++];  
 }  
 else  
 {  
 tmp\_arr[tmp\_index++] = arr[mid++];  
 }  
 }  
 while(start<=center)  
 {  
 tmp\_arr[tmp\_index++] = arr[start++];  
 }  
 while(mid<=end)  
 {  
 tmp\_arr[tmp\_index++] = arr[mid++];  
 }  
   
 for(tmp\_index=start\_backup; tmp\_index<=end; tmp\_index++)  
 {  
 arr[tmp\_index] = tmp\_arr[tmp\_index];   
 }  
 *//while(start\_backup<=end)*  
 *//{*  
 *//arr[start\_backup] = tmp\_arr[start\_backup++];*  
 *//}*  
}  
void sort(double arr[], int start, int end, int n)  
{  
 if(start<end)  
 {  
 int center = (start+end)/2;  
 sort(arr, start, center, n);  
 sort(arr, center+1, end, n);  
 merge(arr, start, center, end, n);  
 }  
}  
void mergeSort(double arr[], int n)  
{  
 int start=0;  
 int end=n-1;  
 sort(arr, start, end, n);   
}  
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 double arr[5]={3.1,2.3,5.6,4.1,1.2};  
 *//BubbleSort(arr, 5);*  
 mergeSort(arr, 5);  
 int i=0;  
 for(i=0; i<5; i++)  
 {  
 cout << arr[i] << " ";  
 }  
 cout << endl;   
 }  
 return 0;  
}  
*/\**  
*start=0, end=1, length=5*  
*start=0, end=2, length=5*  
*start=3, end=4, length=5*  
*start=0, end=4, length=5*  
*5.6 4.1 3.1 2.3 1.2*  
*\*/*  
87、实现双向链表删除一个节点P，在节点P 后插入一  
个节点，写出这两个函数。  
【参考答案】  
// 删除操作  
Status ListDelete\_DuL(DuLinkList &L,inti,Ele mType &e)  
{  
if(!(p=G e tElemP\_DuL(L,i))) return ERROR;  
e=p->data;  
p->prior->next=p->next;  
p->next->prior=p->pror;  
free(p);  
return OK;  
}  
// 插入操作  
Status ListInsert\_DuL(DuLinkList &L,inti,ElemType &e)  
{  
if(!(p=G e tElemP\_DuL(L,i)))  
return ERROR;  
if(!(s=(DuLinkList)malloc(sizeof(DuLNode))))  
return ERROR;  
s->data=e;  
s->pri or =p;  
p-> next -> prior =s;  
p->next=s;  
s->next=p->next->next;  
return OK;  
}  
 88、把一个链表反向。  
【参考答案】从第一个元素开始，ps指向他，将他（  
ps）指向头节点(ps->next = head) ，将ps设为头节点  
（head = ps; ）操作下一个元素（ps= pe->next;）等  
于是依次将每个元素翻到原头节点前面。  
void reverse(test\* head)  
{  
test\* pe = head;  
test\* ps = head->next;  
while(ps)  
{   
pe->next = ps->next;  
ps->next = head;  
head = ps;  
ps =  pe->next;  
}  
}  
typedef struct Node{  
 int val;  
 struct Node \*next;  
}Link;  
   
Link\* reverse(Link \*head)  
{  
 if(head==NULL || head->next==NULL)  
 return head;  
   
 Link \*pcur=head, \*pnew=NULL, \*prev=NULL, \*pnext=NULL;  
 while(pcur)  
 {  
 pnext=pcur->next;  
   
 if(pnext==NULL)  
 {  
 pnew=pcur;  
 printf("pnew's head=%d\n", pnew->val);   
 }  
   
 pcur->next=prev;  
 prev=pcur;  
 pcur=pnext;  
 }  
   
 return pnew;  
}  
   
Link\* reverse2(Link\* head)  
{  
 if(head==NULL || head->next==NULL)  
 return head;  
   
 Link\* prev = head;  
 Link\* pnext = head->next;  
 while(pnext)  
 {  
 prev->next = pnext->next;  
 pnext->next = head;  
 head = pnext;  
 pnext = prev->next;  
 }  
 return head;  
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 Link \*node1 = (Link\*)malloc(sizeof(Link));  
 node1->val=1;  
 node1->next=NULL;  
   
 Link \*node2 = (Link\*)malloc(sizeof(Link));  
 node2->val=2;  
 node2->next=NULL;  
   
 Link \*node3 = (Link\*)malloc(sizeof(Link));  
 node3->val=3;  
 node3->next=NULL;  
   
 Link \*node4 = (Link\*)malloc(sizeof(Link));  
 node4->val=4;  
 node4->next=NULL;  
   
 node1->next=node2;  
 node2->next=node3;  
 node3->next=node4;  
 *//node4->next=node2;*  
   
 Link \*head=node1;   
 printf("原始链表：\n");   
 while(head)  
 {  
 printf("%d ", head->val);  
 head=head->next;  
 }  
 printf("\n");  
   
 head=node1;   
 head = reverse(head);  
 *//head = reverse2(head);*  
 printf("翻转链表：\n");   
 while(head)  
 {  
 printf("%d ", head->val);  
 head=head->next;  
 }  
 printf("\n");  
 }  
 return 0;  
}  
   
*/\**  
*原始链表：*  
*1 2 3 4*  
*pnew's head=4*  
*翻转链表：*  
*4 3 2 1*  
*\*/*  
   
   
89、将二维数组行列元素互换，存到另一个数组中。  
【参考答案】  
#include <stdio.h>  
main()  
{   int a [2][3]={{1,2,3},{4,5,6}};  
int b[3][2],i,j;  
printf("array a :\n");  
for(i=0;i<=1;i++)  
{   for(j=0;j<=2;j++)  
{   printf("%5d",a[i][j]);  
b[j][i]=a[i][j];  
}  
printf("\n");  
}      
printf("array b :\n");  
for(i=0;i<=2;i++)  
{   for(j=0;j<=1;j++)  
printf("%5d",b[i][j]);  
printf("\n");  
}  
}  
*//二维数组行列元素互换，存到另一个数组中*  
void convertArry()  
{  
 int a[2][3]={{1,2,3},{4,5,6}};  
 int b[3][2];  
 int i,j;  
 printf("a:\n");  
 for(i=0; i<2; i++)  
 {  
 for(j=0; j<3; j++)  
 {  
 printf("%d ", a[i][j]);  
 b[j][i]=a[i][j];  
 }  
 printf("\n");  
 }  
 printf("b:\n");  
 for(i=0; i<3; i++)  
 {  
 for(j=0; j<2; j++)  
 {  
 printf("%d ", b[i][j]);  
 }  
 printf("\n");  
 }  
}  
   
   
   
90、输入一行字符，统计其中有多少个单词。  
【参考答案】  
#include <stdio.h>  
main()  
{     
char str i ng[81];  
int i,num=0,word=0;  
char c;  
gets(string);  
for(i=0;(c=string[i])!='\0';i++)  
if(c==' ')    
word=0;  
else if(word==0)  
{     
word=1;  num++;   }  
printf("There are %d word s in the line\n",num);  
}  
*//输入一行字符，统计其中有多少个单词。*  
int words(char \*arr)  
{  
 int words = 0;  
 int prev\_is\_char=0;  
 int pcur\_is\_char=0;  
 while(\*arr != '\0')  
 {  
 if((\*arr>='a' && \*arr <='z') || (\*arr>='A' && \*arr<='Z'))  
 {  
 pcur\_is\_char=1;  
 }  
   
 if(prev\_is\_char==0 && pcur\_is\_char==1)  
 {  
 words++;  
 }  
 prev\_is\_char = pcur\_is\_char;  
 pcur\_is\_char = 0;  
 arr++;  
 }   
   
 return words;  
}  
int words2(char \*arr)  
{  
 int i,num=0,word=0;  
 char c;  
 for(i=0;(c=arr[i])!='\0';i++)  
 {  
 if(c==' ')   
 word=0;  
 else if(word==0)  
 {   
 word=1;   
 num++;   
 }  
 }  
 return num;  
}  
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char \*str="hello, world, china\*\* i love you!!";  
 int count = words(str); *//保证了单词必须是字母*   
 printf("count=%d\n", count);  
 int count2 = words2(str); *//含有"1234"的错认为是单词*   
 printf("count2=%d\n", count2);  
   
 char \*str1="hello, 1234, world, china\*\* i love you!!";  
 count = words(str1); *//保证了单词必须是字母*   
 printf("count=%d\n", count);  
 count2 = words2(str1); *//含有"1234"的错认为是单词*   
 printf("count2=%d\n", count2);  
 }  
 return 0;  
}  
   
*/\**  
*count=6*  
*count2=6*  
*count=6*  
*count2=7*  
*\*/*  
   
   
   
91、写一个内存拷贝函数,不用任何库函数.就是前些时  
候本版讨论的那个问题。  
【参考答案】  
void\* m e mcpy(void\* pvTo, const void\* pvFrom, size\_t size)  
{  
assert((pvTo != NULL) && (pvFrom ! = NULL));  
byte\* pbTo= pvTo;  
byte\* pbFrom = pbFrom;  
while (size-- >  0)  
{  
\*pbTo++ = \*pbFrom++;  
}  
return pvTo;  
}  
   
<https://blog.csdn.net/Wilsonboliu/article/details/7919773>    
void\* my\_memcpy(void\* dest, void\* src,size\_t n)  
{  
 if(dest==NULL || src==NULL)  
 return NULL;  
   
 void\* ret = dest;  
 unsigned char\* str1 = (unsigned char\*)dest;  
 unsigned char\* str2 = (unsigned char\*)src;  
   
 while (n--)  
 {   
 \*str1++ = \*str2++;  
   
 }  
 return ret;  
}  
void\* my\_memmove(void\* dest, void\* src, size\_t n)  
{  
 if(dest==NULL || src==NULL)  
 return NULL;  
   
 void\* ret = dest;  
 unsigned char\* str1 = (unsigned char\*)dest;  
 unsigned char\* str2 = (unsigned char\*)src;  
   
 if(str1 > str2)  
 {  
 while(n--)  
 {  
 \*(str1+n)=\*(str2+n);  
 }  
 }  
 else  
 {  
 while(n--)  
 {  
 \*str1++ = \*str2++;  
 }  
 }  
 return ret;  
}  
   
char \*strcat1(char \*dest, const char \*src) *//将源字符串加const，表明其为输入参数*  
{  
 assert((dest!=NULL) && (src!=NULL));  
 char \*ret = dest;  
 while(\*dest)  
 {  
 dest++;  
 }  
 while(\*dest++=\*src++);  
 return ret;  
}  
char \*strcpy2(char \*dest, const char \*src)  
{  
 assert((dest!=NULL)&&(src!=NULL));  
 char \*ret = dest;  
 while(\*dest++=\*src++);  
 return ret;  
}  
char \*strcpy1(char \*strDest,const char \*strSrc)  
{  
 if((strDest==NULL)||(strSrc==NULL))   
 return NULL;   
   
 char \*ret = strDest; *//[1]*  
 while ((\*strDest++=\*strSrc++)!='\0'); *//[2]*  
 return ret;*//[3]*  
}  
int strcmp1 (const char \*str1,const char \*str2)  
{  
 assert((str1!=NULL) && (str2!=NULL));  
 *//assert((str1 != '\0') && (str2 != '\0'));*  
 while(\*str1 && \*str2 && (\*str1==\*str2))  
 {  
 str1++;  
 str2++;  
 }  
 return \*str1-\*str2;  
}  
   
int strlen1(const char \*str)  
{  
 if(str==NULL)  
 return 0;  
 int len=0;  
 while(\*str++ != '\0')  
 {  
 len++;  
 }  
 return len;  
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 char str1[100]={"i love"};  
 char str2[50]={"China"};  
 *//printf("%s\n",strcat1(str1,str2));*  
 *//printf("%s\n",strcpy1(str1,str2));*  
 *//printf("%d\n",strcmp1(str1,""));*  
 printf("%d\n",strlen1(str2));  
 }  
 if(0)  
 {  
 int arr1[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };  
 int arr2[10] = { 0 };  
 my\_memcpy(arr2, arr1, sizeof(arr1));  
 int i;  
 for (i = 0; i < 10; i++)  
 {  
 printf("%d ", arr2[i]);  
 }  
 printf("\n");  
   
 int arr3[10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };  
 my\_memmove(arr3 + 5, arr3 + 4, 3 \* sizeof(int));  
 for (i = 0; i < 10; i++)  
 {  
 printf("%d ", arr3[i]);  
 }  
 printf("\n");  
 }  
 return 0;  
}  
   
92、有1、2、3 、4个数字，能组成多少个互不相同且  
无重复数字的三位数？都是多少？  
【参考答案】  
#inc lude  "stdio.h"  
#include "conio.h"  
main()  
{  
int i,j,k;  
printf("\n");  
for(i=1;i<5;i++) /\* 以下为三重循环\*/  
for(j=1;j<5;j++)  
for (k=1;k<5;k++)  
{  
if (i!=k&&i!=j&&j!=k) /\* 确保i、j、k三位互不相同\*/  
pr intf("%d,%d,%d\n",i,j,k);  
}  
getch();  
}

93、取一个整数a从右端开始的4～7位。  
【参考答案】  
main()  
{  
unsigned a,b,c,d;  
scanf("%o",&a);  
b=a>>4;  
c=~(~0<<4);  
d=b&c;  
printf("%o\n%o\n",a,d);  
}  
   
*//取一个整数a从右端开始的4～7位*  
void getNumsRight()  
{  
 char str[50]={0};  
 int len=0;  
 unsigned int num=12345678;  
 char val;  
 while(num>0)  
 {  
 val = num%10;  
 num = num/10;  
 str[len]=val+'0';  
 len++;  
 }  
 printf("num str = %s\n", str);  
   
 int i;  
 for(i=3; i<7; i++)  
 {  
 printf("%c", str[i]);  
 }  
 printf("\n");  
}  
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 getNumsRight();  
 }  
 return 0;  
}  
   
*/\**  
*num str = 87654321*  
*5432*  
*\*/*  
   
   
94、打印出杨辉三角形（要求打印出10行如下图）。  
【参考答案】  
main()  
{  
int i,j;  
int a [10][10];  
printf("\n");  
for(i=0;i<10;i++)  
{  
a[i][0]=1;  
a[i][i]=1;  
}  
for(i=2;i<10;i++)  
for(j=1;j<i;j++)  
a[i][j]=a[i-1][j-1]+a[i-1][j];  
for(i=0;i<10;i++)  
{  
for(j=0;j<=i;j++)  
printf("%5d",a[i][j]);  
printf("\n");  
}  
getch();   
}  
void yanghuiTriangle(int M)  
#define PYRAMID *// 金字塔，会额外填充空格*  
*// #define REVERSE // 反向再来一次，得到菱形*  
{  
 int a [M][M], i, j; *// 二维数组和循环变量，a[行][列]*  
 for (i = 0; i<M; i++) *// 每一行*  
 {  
#ifdef PYRAMID  
 for (j = 0;j <= M-i; j++)   
 printf (" ");  
#endif *// 填充结束*  
   
 for (j = 0; j <= i; j++) *// 赋值打印*  
 {  
 if(i == j || j == 0)  
 {  
 a[i][j] = 1;  
 }  
 else  
 {  
 a[i][j] = a[i-1][j] + a[i-1][j-1]; *// 使用上一行计算*  
 }  
 printf("%4d", a[i][j]);  
 }  
 printf("\n");  
 }  
   
   
void yanghui(int M)  
{  
 int i,j;  
 int a [M][M];  
   
 printf("\n");  
   
 *//for(i=0;i<10;i++)*  
 *//{*  
 *//a[i][0]=1;*  
 *// a[i][i]=1;*  
 *//}*  
   
 for(i=0;i<M;i++)  
 {  
 for(j=0;j<=i;j++)  
 {  
 if(i==j || j==0)  
 a[i][j] = 1;  
 }  
 }  
   
 for(i=2; i<M; i++)  
 for(j=1; j<i; j++)  
 a[i][j] = a[i-1][j-1] + a[i-1][j];  
   
 for(i=0; i<M; i++)  
 {  
 *//for(j = 0;j <= M-i; j++) //添加此行为金字塔，不添加为直角三角形*  
 *//printf (" ");*  
 for(j=0; j<=i; j++)  
 printf("%5d", a[i][j]);  
 printf("\n");  
 }  
   
}  
   
int main(int argc, char \*argv[])  
{  
 if(1)  
 {  
 yanghui(10);  
 }  
 return 0;  
}  
   
  
   
95、实现strcmp 函数。  
100 条经典 C语言笔试题目  
96、写一个函数，求一个字符串的长度，在main函数  
中输入字符串，并输出其长度。  
【参考答案】  
main()  
{  
intlen;  
char \*s tr[20];  
printf("please input a string:\n");  
scanf("%s",str);  
len=length(str);  
printf("the s tring has %d characters.",len);  
getch();  
}  
length(p)  
char \*p;  
{  
int n ;  
n=0;  
while(\*p!='\0')  
{  
n++;  
p++;  
}  
return n;  
}  
97、809\*??=800\*??+9\*??+1  其中??代表的两位数  
,8\*?? 的结果为两位数，9\*??的结果为3位数。求??代  
表的两位数，及809\*??后的结果。  
【参考答案】output(long b,long i)  
{  
printf("\n%ld/%ld=809\*%ld+%ld",b,i,i,b%i);  
}  
main()  
{  
long int a ,b,i;  
a=809;  
for(i=10;i<100;i++)  
{  
b=i\*a+1;  
if(b>=1000&&b<=10000&&8\*i<100&&9\*i>=100)  
output(b,i);  
}  
getch();  
}  
   
98、某个公司采用公用电话传递数据，数据是四位的整数，在  
传递过程中是加密的，加密规则如下：每位数字都加上5, 然后  
用和除以10的余数代替该数字，再将第一位和第四位交换，第  
二位和第三位交换。  
【参考答案】  
main()  
{  
int a ,i,aa[4],t;  
scanf( " % d " ,&a) ;  
aa[0]=a %10;  
aa[1]=a %100/10;  
aa[2]=a %1000/100;  
aa[3]=a /1000;  
for(i=0 ; i<=3 ;i++)  
{  
aa [i]+=5;  
aa[i] %=10;  
}  
for(i=0;i<=3/2;i++)  
{  
t=aa[i];  
aa[i]= aa[3- i];  
aa[3- i]= t ;  
}  
for(i=3;i>=0;i--)  
prin tf("% d ", a [i]);  
getch();  
}  
99、计算字符串中子串出现的次数。  
【参考答案】  
main()  
{  
char str1[20],str2[20],\*p1,\*p2;  
int sum=0;  
printf("please input two strings\n");  
scanf("%s%s",str1,str2);  
p1=str1;p2=str2;  
while(\*p1!='\0')  
{  
if(\*p1==\*p2)  
{  
while(\*p1==\*p2&&\*p2!='\0')  
{  
p1++;  
p2++;  
}  
}  
else  
p1++;  
if(\*p2=='\0')  
sum++;  
p2=str2;  
}  
printf("%d",sum);  
getch();  
}   
 100 、有两个磁盘文件A和B, 各存放一行字母，要求把  
这两个文件中的信息合并（按字母顺序排列），输出  
到一个新文件C 中。  
【参考答案】  
100 条经典 C语言笔试题目  
main()  
{  
FILE \*fp;  
in t i,j,n,ni;  
char c[160],t,ch;  
if((fp=fopen("A","r "))==NULL)  
{  
printf("file A cannot  be opened\n");exit(0);  
}  
printf("\nA contents are :\n");  
for(i=0;(ch=fgetc(fp))!=EOF ; i++)  
{  
c[i]=ch;pu tchar(c[i]);  
}  
fclose(fp);  
ni=i;  
if((fp=fopen("B","r "))==NULL)  
{  
printf("file B cannot  be opened\n");exit(0);  
}  
printf("\nB contents are :\n");  
for(i=0;(ch=fgetc(fp))!=EOF ; i++)  
{  
c[i]=ch;pu tchar(c[i]);  
}  
fclose(fp);  
n=i;  
for(i=0;i<n;i++)  
for(j=i+1;j<n;j++)  
if(c[i]>c[j]){t=c[i ];c[i]=c[j];c[j]=t;}  
printf("\nC file is:\n");  
fp =f open("C","w");  
for(i=0;i<n;i++){putc(c[i ],fp);putchar(c[i]);}  
fclose(fp);  
getch();  
}