**1. C语言中，修饰符volatile含义是什么？其应用场合有哪些？**   
volatile提醒编译器它后面所定义的变量随时都有可能改变，因此编译后的程序每次需要存储或读取这个变量的时候，都会直接从变量地址中读取数据。如果没有volatile关键字，则编译器可能优化读取和存储，可能暂时使用寄存器中的值，如果这个变量由别的程序更新了的话，将出现不一致的现象。

**2. 请问TCP/IP协议分为哪几层？FTP协议在哪一层？**   
ISO/OSI的参考模型共有7层，由低层至高层分别为：物理层、数据链路层、网络层、传输层、会话层、表示层、应用层   
物理层：在物理媒体上传输原始的数据比特流。   
数据链路层：将数据分成一个个数据帧，以数据帧为单位传输。有应有答，遇错重发。   
网络层：将数据分成一定长度的分组，将分组穿过通信子网。   
传输层：提供不具体网络的高效、经济、透明的端到端数据传输服务。   
会话层：进程间的对话也称为会话，会话层管理不同主机上各进程间的对话。   
表示层： 为应用层进程提供格式化的表示和转换数据服务。   
应用层：提供应用程序访问OSI环境的手段。

应用层：TELNET、FTP、TFTP、SMTP、SNMP、HTTP、BOOTP、DHCP、DNS   
表示层：   
文本：ASCII，EBCDIC   
图形：TIFF，JPEG，GIF，PICT   
声音：MIDI，MPEG，QUICKTIME   
会话层：NFS、SQL、RPC 、X-WINDOWS、ASP（APPTALK会话协议）、SCP   
传输层：TCP、UDP、SPX   
网络层：IP、IPX、ICMP、RIP、OSPF(Open Shortest Path First开放式最短路径优先)   
数据链路层：SDLC、HDLC、PPP、STP（Spanning Tree Protocol）、帧中继   
物理层：EIA/TIA RS-232、EIA/TIA RS-449、V.35、RJ-45

**3. 在网络应用中，函数htons,htonl,ntohs,ntohl的作用是什么？**   
uint32\_t htonl(uint32\_t hostlong);//32位的主机字节序转换到网络字节序   
uint16\_t htons(uint16\_t hostshort);//16位的主机字节序转换到网络字节序   
uint32\_t ntohl(uint32\_t netlong);//32位的网络字节序转换到主机字节序   
uint16\_t ntohs(uint16\_t netshort);//16位的网络字节序转换到主机字节序   
(皆为大小端的改变)

**4. C语言中static函数与普通函数的区别是什么？**   
静态函数   
在函数的返回类型前加上关键字static，函数就被定义成为静态函数。   
函数的定义和声明默认情况下是extern的，但静态函数只是在声明他的文件当中可见，不能被其他文件所用。   
定义静态函数的好处：   
<1> 其他文件中可以定义相同名字的函数，不会发生冲突   
<2> 静态函数不能被其他文件所用。   
局部静态变量   
在局部变量之前加上关键字static，局部变量就被定义成为一个局部静态变量。   
1）内存中的位置：静态存储区   
2）初始化：未经初始化的全局静态变量会被程序自动初始化为0（自动对象的值是任意的，除非他被显示初始化）   
3）作用域：作用域仍为局部作用域，当定义它的函数或者语句块结束的时候，作用域随之结束。

全局静态变量   
在全局变量之前加上关键字static，全局变量就被定义成为一个全局静态变量。   
1）内存中的位置：静态存储区（静态存储区在整个程序运行期间都存在）   
2）初始化：未经初始化的全局静态变量会被程序自动初始化为0（自动对象的值是任意的，除非他被显示初始化）   
3）作用域：全局静态变量在声明他的文件之外是不可见的。准确地讲从定义之处开始到文件结尾。

好处：   
定义全局静态变量的好处：   
<1>不会被其他文件所访问，修改   
<2>其他文件中可以使用相同名字的变量，不会发生冲突

**5. 请实现内存复制函数void memcpy(void \*dst const void \*src,int size)**

void \*memcpy(void \*dest, const void \*src, unsigned int size)

{

if(dest==NULL || src==NULL)

{

return dest;

}

void \*temp = dest;

while(size--)

{

\*(char \*)dest = \*(char \*)src;

dest = (char \*)dest + 1;

src = (char \*)src + 1;

}

return temp;

}

6、进程间通信的方式有

1. 管道（pipe）,流管道(s\_pipe)和有名管道（FIFO）
2. 信号（signal）
3. 消息队列
4. 共享内存
5. 信号量
6. 套接字（socket)

**7、IPv4把所有的IP地址分为A、B、C、D、E五类。请写出B类地址和C类地址的范围和掩码，D类地址的用途是什么？**   
A类IP地址范围：0.0.0.0到127.255.255.255   
B类IP地址范围：128.0.0.0到191.255.255.255   
C类IP地址范围：192.0.0.0到223.255.255.255   
D类IP地址范围：224.0.0.0到239.255.255.255   
E类IP地址范围：224.0.0.0到254.255.255.255

D类地址用于多点播送   
E类地址保留，仅作实验和开发用

全零（“0．0．0．0”）地址指任意网络。   
全“1”的IP地址（“255．255．255．255”）是当前子网的广播地址。

**8、请简述在linux中多进程程序和多线程程序的区别。** 

9、写一个标准的宏MINX，这个宏输入两个参数并返回较小的一个?

#define MIN(A, B) ((A)>(B)? (B) : (A))

10、什么是MMU,MMU的作用

MMU是Memory Management Unit(内存管理单元)   
1）虚拟内存。有了虚拟内存，可以在处理器上运行比实际物理内存大的应用程序。为了使用虚拟内存，操作系统通常要设置一个交换分区（通常是硬盘），通过将不活跃的内存中的数据放入交换分区，操作系统可以腾出其空间来为其它的程序服务。虚拟内存是通过虚拟地址来实现的。   
2）内存保护。根据需要对特定的内存区块的访问进行保护，通过这一功能，我们可以将特定的内存块设置成只读、只写或是可同时读写

11、**关键字volatile有什么含义，并给出几个应用场合。**   
volatile的本意是“易变的” 由于访问寄存器的速度要快过RAM,所以编译器一般都会作减少存取外部RAM的优化，但有可能会读脏数据。当要求使用volatile 声明的变量的值的时候，系统总是重新从它所在的内存读取数据，即使它前面的指令刚刚从该处读取过数据。而且读取的数据立刻被保存。   
一般说来，volatile用在如下的几个地方：   
1、中断服务程序中修改的供其它程序检测的变量需要加volatile；   
2、多任务环境下各任务间共享的标志应该加volatile；   
3、存储器映射的硬件寄存器通常也要加volatile说明，因为每次对它的读写都可能由不同意义；

**12、“栈 stack”和“堆 heap”有什么区别，写两个函数分别在栈上和堆上分配内存，在堆上分配内存应注意哪些问题?**

1.申请方式   
stack:   
由系统自动分配。 例如，声明在函数中一个局部变量   
int b; 系统自动在栈中为b开辟空间   
heap:   
需要程序员自己申请，并指明大小，在c中malloc函数   
p1 = (char \*)malloc(10);   
2.申请后系统的响应   
栈：   
只要栈的剩余空间大于所申请空间，系统将为程序提供内存，否则将报异常提示栈溢出。   
堆：   
首先应该知道操作系统有一个记录空闲内存地址的链表，当系统收到程序的申请时，   
会遍历该链表，寻找第一个空间大于所申请空间的堆结点，然后将该结点从空闲结点链表   
中删除，并将该结点的空间分配给程序.   
3.申请大小的限制   
栈：   
栈顶的地址和栈的最大容量是系统预先规定好的，是一块连续的内存的区域，在WINDOWS下，栈的大小是2M ，如果申请的空间超过栈的剩余空间时，将提示overflow。因此，能从栈获得的空间较小。 （有限内存）   
堆：   
堆是向高地址扩展的数据结构，是不连续的内存区域。（自己分配）

void fun(void)

{

int a[10];

int \*p = (int \*)malloc(10\*sizeof(int));

if(p == NULL)

{

return;

}

}

**13、请指出下面程序的错误**

unsigned char i = 10;

char b = 100;

char\* p;

void f(void)

{

While(–-i>=0)

{

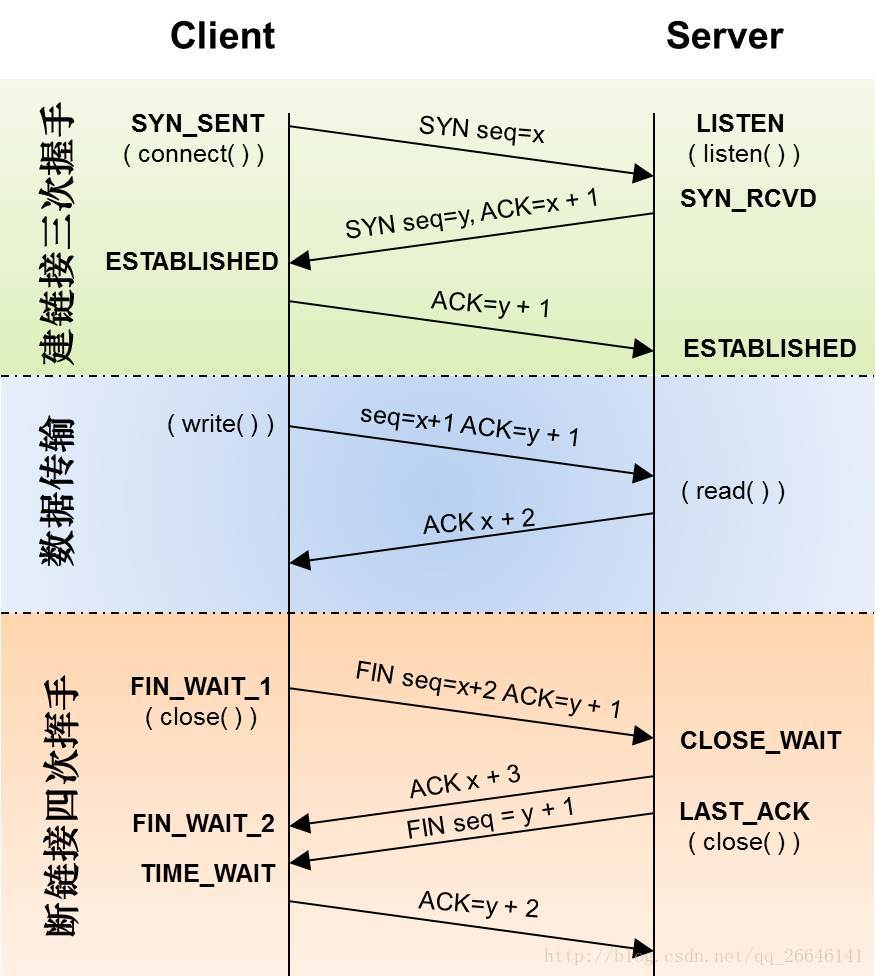
\*(p++) = b;

}

}

p没有分配空间   
i是无符号型，自减操作会引发死循环

14、**TCP/IP通信建立的过程怎样，端口有什么作用?**



端口作用：（连接作用）就是让运输层的各种应用进程都能将其数据通过端口向下交付给运输层,以及让运输层知道应当将其报文段的数据向上通过端口交付给应用层相应的进程