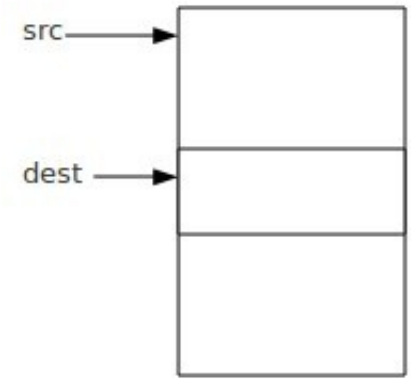
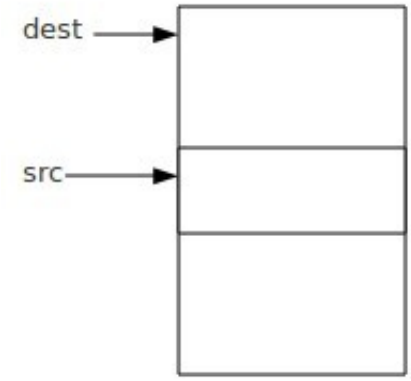
Linux库memcpy函数实现

memcpy函数在面试中很容易被问到如何去实现。memcpy函数是内存拷贝函数，用于将一段内存空间数据拷贝到另一段内存空间中，但是它和memmove函数不同的是，它对内存空间有要求的，dest和src所指向的内存空间不能重叠，否则的数据是错误的。例如：



src所指向的内存空间后面部分数据被新拷贝的数据给覆盖了，所以拷贝到最后，数据肯定不是原来的数据。



如果内存空间像上图所示，就不会导致数据拷贝错误这种情况，也就说你在使用这个函数之前，还需要做一个判断，如果dest < src你才能使用这个函数，不过完全没有必要，你直接使用另外一个函数memmove函数就可以了。基于上面的一些原因，不建议在程序中使用memcpy函数，而直接用memmove代替。
函数原型：

```
#include<string.h>

void *memcpy(void *s1, const void *s2, size_t n);
```

The memcpy() function copies n bytes from the object pointed to by s2 into the object pointed to by s1. If copying takes place between objects that overlap, the behaviour is undefined.
它的实现：

```
#include<string.h>

void *
memcpy(void *s1, const void *s2, register size_t n)
{
 register char *p1 = s1;
 register const char *p2 = s2;

 if (n) {
  n++;
  while (--n > 0) {
   *p1++ = *p2++;
  }
 }
 return s1;
}
```

使用memcpy函数需要注意：

1. s1和s2所指内存区域不能重叠，函数返回指向s1的指针；
2. strcpy和memcpy主要有以下3方面的区别。
   1)、复制的内容不同。strcpy只能复制字符串，而memcpy可以复制任意内容，例如字符数组、整型、结构体、类等。
   2)、复制的方法不同。strcpy不需要指定长度，它遇到被复制字符的串结束符"₩0"才结束，所以容易溢出。memcpy则是根据其第3个参数决定复制的长度。
   3)、用途不同。通常在复制字符串时用strcpy，而需要复制其他类型数据时则一般用memcpy
3. 如果目标数组destin本身已有数据，执行memcpy（）后，将覆盖原有数据（最多覆盖n）。如果要追加数据，则每次执行memcpy后，要将目标数组地址增加到你要追加数据的地址。
4. s1和s2都不一定是数组，任意的可读写的空间均可。