1. \*source和\*dest都是指向字符串中的字符的。当\*source++时，将该字符赋值到dest指向的字符串中的字符，同时两个指针都向后移动一个字符。当\*source指向‘\0’（即串结束标志）时，循环结束。每次指针（目标和原始指针）都向后移动1个字节，指向内容移向字符串的后一个字节。
2. temp存储了字符串的首地址，如果改为dest，则从字符串最后的字节开始读取，并越界读取其他内存，导致乱码。
3. 指针str1的目标是’\0’字符
4. 防止指针变量 str1 和str2指向发生变化，导致传入的全局指针变量随之改变。
5. 本循环判断的是指针p，q指向的字符都不是‘\0’且p，q指向的字符相同。即str1和str2指向的字符串长度不同，结束；\*p，\*q指向的字符不同也会结束。
6. ‘\0’的ascii码是0，起到使循环结束的作用（无论是直接while读取到它还是赋值时读取到它）。
7. 否 这不是python
8. 因为字符数字是以ascii码形式储存的，减去零的值才是它的显示值。
9. 因为我们在这个函数中是要改变str指向的字符串的长度的。
10. 在delete[] str;之后，str变为野指针，因此需要str = NULL防止指针越界。
11. 因为dest要重新申请一个指针数组，用于存放5个字符串。
12. 循环用于确定指针的指向，与前一个指针相差NUM个元素。
13. strncpy函数限定了拷贝字符串的长度，防止超长字符串破坏内存中的其他数据。
14. 防止超限字符串末尾不带’\0’
15. 为了能删除str指针后再赋值其NULL，使其完成释放堆空间后指针改成NULL的约定。
16. 在排序后，指针可能不是第一个指向首地址，而是最小的指向首地址。因此需要进行一次循环比较找到最小的以便释放
17. 释放三个函数头new所在堆空间中申请的动态指针数组
18. 防止指向堆空间的str指针变成野指针。也是为了守护“约定”，将没有指向特定动态字符串数组的指针的值设置为NULL
19. 变了，应为两个函数都是引用传递，不过free完一个循环后值是一样的，都是NULL
20. sizeof(strA) : 300 Bytes, sizeof(\*strA) : 20 Bytes, n1 = 15

sizeof(strB) : 120 Bytes, sizeof(\*strB) : 8 Bytes, n2 = 15

sizeof(strC) : 8 Bytes

sizeof(strD) : 8 Bytes

sizeof(strE) : 8 Bytes

21.strA 是一个二维字符数组，其中的字符串是在栈上分配的，它们的生命周期与 TestSortStrings 函数的执行周期相同。

strB 是一个指针数组，其中的每个指针指向一个字符串常量。这些字符串常量通常存储在常量池中，它们的生命周期在整个程序运行期间都存在。

strC、strD、strE 是指针数组，它们的指针指向动态分配的字符串数组，这些字符串数组存储在堆上。

图不会画

22.bubbleB是交换指针数组的函数，不能交换普通二维数组

23.bubbleA有两个，通过函数重载来适应不同实参

但strcpy交换出现了修改指针指向的常量的错误

24.strC是getstring（）函数在堆区创造的，执行BubbleB时是改变了指针dest的指向，因此strc[0]会被改变

25.strC是getstring0函数创建的，是连续存放的，因此strC在删除时只需删除两次，不用n+1次，且需要找到存放最小指针的strC头

26.使用了bubbleA第二个重载函数，对其排序的方式是改变strC[]每一个指针指向的内容，因此strc[0]不变，但其指向的内容会改变。

27.strD是getstrings1函数创建的，其中每个字符串的长度相等，因此既可以用bubbleA交换指针指向的字符串，也可以交换指针

28.strE是通过getstrings2创建的，其每个字符串的长度是自身长度，因此bubbleA函数简单的交换字符串内容会导致字符串越界的情况出现

29.参数默认是零，因此写成false和不写没有区别

30.由于strC[0]并没有被改变，因此n=1，bool=0执行的是delete[] strs[0],是可以释放字符串的。