排序的实现

限制

- 1000 ms
- 32768 KB

我们刚刚在课程中学习了简单的冒泡排序和选择排序，在这一节，你将被给予 10 个乱序输入的整数。你需要（任选一种排序方法）将它们从大到小进行排序后输出。

输入格式

测评机会反复运行你的程序。每次程序运行时，你的程序仅需输入 10 个由空格分隔的整数 $Number_i$（$1 \leq Number_i \leq 1000$），（其中 $1 \leq i \leq 10$）。

输出格式

输出为一行，包括排序后的 10 个输入整数，每两个整数之间有一个空格，最后一个整数后面没有空格。

样例输入 1

```
1 2 3 4 5 6 7 8 9 10
```

样例输出 1

```
10 9 8 7 6 5 4 3 2 1
```

样例输入 2

```
2 3 1 9 5 4 4 3 3 2
```

样例输出 2

```
9 5 4 4 3 3 3 2 2 1
```

提示信息

你的程序会输入各种不同的符合描述的数据（所以你需要用`scanf`输入）。它之后会将程序的输出和正确答案进行比对，完全一样即为正确。如果你的程序对任何一组输入都给出了正确的输出，你就会通过这道题目的测试。

请注意，不要让你的程序输出任何多余的内容，否则测评机都会给出“运行结果错误”的提示。

如果你遇到了格式错误，很有可能是因为你没有做到“最后一个整数后面没有空格”。要实现这个要求的一种方式是合理判断我们是否到达最后一个数。这往往会通过使用循环控制变量的值进行判断。例如下面这个例子中，我们通过判断 `j != 9` 的方式再未达到最后一个输出数字时，多输出一个空格。

```
for (j = 0; j < 10; j++) {
    printf("%d", j);
    if (j != 9) {
        printf(" ");
    }
}
```

具体的设计这次要交给你了喔！如果觉得实在困难，可以参考下下面用于冒泡排序的代码片段。不过请注意，`swap` 函数需要你自己实现喔。

```
for (j = 0; j < 5; j++) {
    for (i = 0; i < 4 - j; i++) {
        swap(a[i], a[i + 1]);
    }
}
```