Sorting: HeapSort and QuickSort

Description In this assignment you will implement two different algorithms for sorting elements in an array: Heapsort and Quicksort. In Quicksort, use a *random* element of the array as your pivot.

For the test of your codes, use "GradeQuick" and "GradeHeap" in your working directory. You also need to be sure the directory "testfiles" is in your working directory and that its content have not been modified.

For Heapsort, your execution file's name must be "HeapSort.exe." Likewise, for Quicksort, your execution file's name must be "QuickSort.exe." As in the previous lab, perhaps you will need to use chmod to change the execution permissions of "GradeQuick" and "GradeHeap". If so, you can just type: "chmod 700 GradeQuick" and "chmod 700 GradeHeap".

Now if you run "./GradeQuick" and your implementation of Quicksort is correct, you will see the following messages.

```
Testing QuickSort.exe:
Correct for 1 th example.
Correct for 2 th example.
Correct for 3 th example.
Correct for 4 th example.
Correct for 5 th example.
Correct for 6 th example.
Correct for 7 th example.
Correct for 8 th example.
Correct for 9 th example.
Correct for 10 th example.
10/10
```

Similarly, you will have to run "./GradeHeap" for testing Heapsort.

You can find a log of the execution in the file named "result".

You are required to compile your codes in front of the TA and answer any questions in order to show that you fully understand what your wrote. If you want to test your code for just one test file, you can try: "./QuickSort.exe < ../testfiles/t1". Of course you can also create your own test input file(s).

The included simple "Makefile" will compile both QuickSort.cpp and HeapSort.cpp. If you type "make clean", then QuickSort.exe and HeapSort.exe will be deleted.

Important Reminder: Never change the grading scripts of the files under the "testfiles" folder. If you do so, it will be considered as SERIOUS CHEATING.

Input structure The input starts with an integer number which indicates the number of elements (integers) to be sorted. Then, the elements follow, one per line.

Output structure Output the sorted sequence one element per line. Do *not* insert spaces at the beginning or at the end of any element.

Examples of input and output

Example:
Input

```
6
3
6
23
76
4
56
```

Output

```
3
4
6
23
56
76
```

Submission As usual, before the posted deadline, submit a .zip or zipped tar archive of your program through the assignments page of CatCourses. Please use your UCMNetID as the filename for the zipped archive. Be careful since CatCourses strictly enforces the assignment deadline. Recall that submission alone is not enough, you must present your work to your TA before submission.