Radixsort

Description In this assignment you will implement RadixSort.

For testing your code you will use "Grade03", which should be in your working directory. Sub-directory "testfiles" contains files t1, t2, ..., t5 and o1, o2, ..., o5: for each testfile tx, the right answer is in file ox. After you compile your code, your execution file's name must be "RadixSort.exe". Perhaps you may need to change the permission of "Grade03" as explained in previous labs.

If you run "./GradeMe03" and your code is correct, you will see the following messages:

Correct for 1 th example.
Correct for 2 th example.
...

Input structure You are going to apply RadixSort to sort vectors. The input starts with an integer number which indicates the number of vectors to be sorted. Then vectors follow, one vector per line. Each vector consists of 10 numbers where each number has a value in $\{0, 1, 2, 3\}$. Entries of a vector are separated by a space.

Output structure Output the sorted sequence of vectors, one per line. Vector i must appear before vector j in your output if and only if for some $d \in \{1, 2, ..., 10\}$, vector i is smaller than or equal to vector j on the dth entry, and the two vectors are equal for any of the first $d-1$ entries.

Examples of input and output

Input

```
5
3 3 3 3 3 2 2 2 2 2
2 3 2 2 2 2 2 2 2 2
1 3 0 0 2 1 0 0 0 0
1 3 0 0 2 2 0 0 0 0
2 3 2 1 2 2 2 2 2 2
```

Output

```
1;3;0;0;2;1;0;0;0;0;
1;3;0;0;2;2;0;0;0;0;
2;3;2;1;2;2;2;2;2;2;
2;3;2;2;2;2;2;2;2;2;
3;3;3;3;3;2;2;2;2;2;
```

More precisely, the above output example has 6 lines since a "cout << endl;" call was made at the end of each of the first 5 lines; those are the only white characters.

You can see the grader's output files, o1, o2, $\cdots$ in the testfiles folder for more details. The above toy example is in t0, perhaps you want to try:

./RadixSort.exe < ./testfiles/t0 > my0
diff my0 ./testfiles/o0

This will show the difference between your output and the output file that the grading toolkit uses. It is always a good idea to start with a simple example when testing your code.

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.

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.