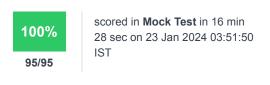


Mock Test > doliatius@protonmail.com

Full Name: Musab Oguz Email: doliatius@protonmail.com Test Name: **Mock Test** Taken On: 23 Jan 2024 03:51:50 IST Time Taken: 16 min 28 sec/ 30 min Linkedin: https://www.linkedin.com/in/musab-oguz-68990a200/ Invited by: Ankush Invited on: 23 Jan 2024 03:51:03 IST Skills Score: Tags Score: Algorithms 95/95 Arrays 95/95 Core CS 95/95 95/95 Data Structures 95/95 Easy Sorting 95/95 Strings 95/95



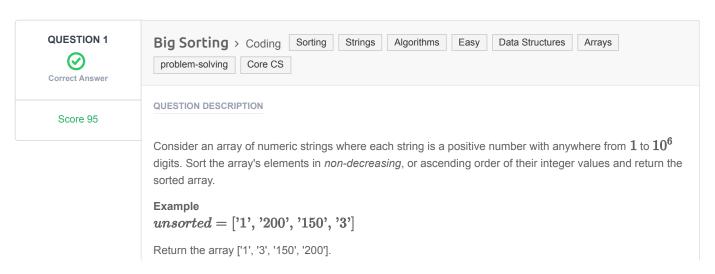
Recruiter/Team Comments:

No Comments.



95/95

problem-solving



Function Description

Complete the bigSorting function in the editor below.

bigSorting has the following parameter(s):

• string unsorted[n]: an unsorted array of integers as strings

Returns

• string[n]: the array sorted in numerical order

Input Format

The first line contains an integer, n, the number of strings in unsorted. Each of the n subsequent lines contains an integer string, unsorted[i].

Constraints

- $1 \le n \le 2 \times 10^5$
- Each string is guaranteed to represent a positive integer.
- There will be no leading zeros.
- The total number of digits across all strings in unsorted is between 1 and 10^6 (inclusive).

Sample Input 0

```
6
31415926535897932384626433832795
1
3
10
3
5
```

Sample Output 0

```
1
3
3
5
10
31415926535897932384626433832795
```

Explanation 0

The initial array of strings is

unsorted = [31415926535897932384626433832795, 1, 3, 10, 3, 5]. When we order each string by the real-world integer value it represents, we get:

$$1 \leq 3 \leq 3 \leq 5 \leq 10 \leq 31415926535897932384626433832795$$

We then print each value on a new line, from smallest to largest.

Sample Input 1

```
8
1
2
100
12303479849857341718340192371
3084193741082937
3084193741082938
111
200
```

Sample Output 1

```
1
2
100
111
200
3084193741082937
3084193741082938
12303479849857341718340192371
```

CANDIDATE ANSWER

Language used: C++14

```
2 /*
 3 * Complete the 'bigSorting' function below.
   * The function is expected to return a STRING ARRAY.
* The function accepts STRING ARRAY unsorted as parameter.
7 */
9 bool sortlength(const string& a, const string& b) {
10    if (a.size() == b.size()) {
          return a < b;
     return a.size() < b.size();
14 }
16 vector<string> bigSorting(vector<string> unsorted) {
     vector<string> sorted(unsorted.size());
     sort(unsorted.begin(), unsorted.end(), sortlength);
      for (unsigned int i = 0; i<unsorted.size()-1; i++){
          if (unsorted.at(i).size() == unsorted.at(i+1).size() &&
23 unsorted.at(i)>unsorted.at(i+1)){
              swap(unsorted.at(i), unsorted.at(i+1));
          }
      }
      return unsorted;
29 }
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	Success	0	0.0374 sec	8.81 KB
Testcase 2	Medium	Hidden case	Success	10	0.0063 sec	8.82 KB
Testcase 3	Medium	Hidden case	Success	10	0.0252 sec	9.91 KB
Testcase 4	Hard	Hidden case	Success	15	0.1391 sec	10.9 KB
Testcase 5	Hard	Hidden case	Success	15	0.0405 sec	10.7 KB
Testcase 6	Hard	Hidden case	Success	15	0.0249 sec	10.2 KB
Testcase 7	Hard	Hidden case	Success	15	0.1214 sec	12.8 KB
Testcase 8	Hard	Hidden case	Success	15	0.0936 sec	26.1 KB
Testcase 9	Easy	Sample case	Success	0	0.0096 sec	8.77 KB

PDF generated at: 22 Jan 2024 22:39:59 UTC