Bigger is Greater

by Bidhan

Editorial by dheeraj

We can find the next largest lexicographic string for a given string S using the following step.

Iterating over every character, we will get the last value i (starting from the first character) that satisfies the given condition

```
S[i] < S[i + 1]
```

Now, we will get the last value j such that

```
S[i] < S[j]
```

We now interchange S[i] and S[j]. And for every character from $i + 1$ till the end, we sort the characters. i.e.,

```
sort(S[i+1]..S[len(S) - 1])
```

The given string is the next largest lexicographic string of S.

One can also use *next_permutation* function call in cpp.

Set by Bidhan

Problem Setter's code:

C++

```cpp
#include <bits/stdc++.h>
using namespace std;

int main(){
    int test;
    cin >> test;
    while(test--) {
        string inp;
        cin >> inp;
        if (next_permutation(inp.begin(), inp.end()) == false ) {
            cout << "no answer" << endl;
        }
        else {
            cout << inp << endl;
        }
    }
    return 0;
}
```

Statistics

Difficulty: Medium
Time Complexity: O(N)
Required Knowledge: Lexicographic Strings
Publish Date: Sep 12 2014

Originally featured in CodeSprint India 2014 Qualification Round 1

Tested by dheeraj

Problem Tester's code:

Python 2

```
t = input()
for _ in range(t):
    word = list(raw_input().strip())
    start = -1
    for i in xrange(0, len(word) - 1):
        if word[i] < word[i + 1]:
            start = i

    if start == -1:
        print "no answer"
        continue

    end = -1
    for j in xrange(start + 1, len(word)):
        if word[start] < word[j]:
            end = j

    word[start], word[end] = word[end], word[start]
    a = word[start + 1:]
    a.sort()

    for j in xrange(start + 1, len(word)):
        word[j] = a[j - start - 1]

    print "".join(word)
```