1038. Recover the Smallest Number (30)

时间限制

400 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a collection of number segments, you are supposed to recover the smallest number from them. For example, given {32, 321, 3214, 0229, 87}, we can recover many numbers such like 32-321-3214-0229-87 or 0229-32-87-321-3214 with respect to different orders of combinations of these segments, and the smallest number is 0229-321-3214-32-87.

**Input Specification:**

Each input file contains one test case. Each case gives a positive integer N (<=10000) followed by N number segments. Each segment contains a non-negative integer of no more than 8 digits. All the numbers in a line are separated by a space.

**Output Specification:**

For each test case, print the smallest number in one line. Do not output leading zeros.

**Sample Input:**

5 32 321 3214 0229 87

**Sample Output:**

22932132143287

[提交代码](https://www.patest.cn/contests/pat-a-practise/1038)

这道题一开始看是比较懵逼的，但是仔细一想以后发现是有规律的，就是对于成型数组的每一个位数，要取最小的一个位数，那么我们对于所有数字进行对比，最方便的就是利用特殊的comp函数进行排序的方法；

排序的规则是：

对于一个string a,b,一位一位进行比较，若不同，则该位上比较小的那个优先级高，若相同，则继续比较下一位，若比较的值超过了其中长度较小的一个的范围，那么将这个值移到这个string的开头在进行比较，因为其中一位选出后，那么次于它的一个必然会接到它的尾部，那此时较长的那个比较位置即为较短的那个除余以后的值的位置（因为前面的长度都相等，比如32X,和32，那结果要不就是32X32 要不就是 3232X,那此时就应该是X和32的3比较大小了），然后再选择最优的那个。

BTW，浙大大佬的代码比较是这么写的：

Return a+b<b+a;

Orz。

还有这道题的输出是有坑的……哎其实也算是自己傻吧

即：对于先导0不能输出，那么应该先找到该输出的第一个先导0，并标注它的位置，然后利用标注初始值判断大小是否全是先导0（没错是有这样的数据的）

然后再输出先导0后面的字符串。

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

#include<map>

#include<cstring>

#include<climits>

#include<cmath>

#include<fstream>

#define MAX 1001

#define LL long long

using namespace std;

vector<string>list;

int comp(string a, string b)

{

if (a == b) return a>b;

int alen = a.size(), blen = b.size();

int longth = max(alen, blen);

for (int i = 0; i < longth; i++)

{

if (a[i%alen] != b[i%blen])

{

return a[i%alen] < b[i%blen];

}

}

}

int main()

{

int num;

cin >> num;

list.resize(num);

for (int i = 0; i < num; i++)

{

cin >> list[i];

}

sort(list.begin(), list.end(), comp);

string result;

for (int i = 0; i < list.size(); i++)

{

result += list[i];

}

int pos=-1;

for (int i = 0; i < result.size(); i++)

{

if (result[i] != '0')

{

pos = i;

break;

}

}

if (pos==-1)

{

cout << "0" << endl;

}

else

cout << result.substr(pos);

}