Laboratory work #4

Student: CAO Xinyang  
Student ID: 20321308  
Timus Name: hduads2022\_20321308

Mail: c.x\_yang@foxmail.com

Problem #1322

Screenshot from Timus:



Explanation of algorithm:

The first column of the matrix is obtained by sorting the array of the last known column of the matrix. And then sort the last column of the matrix as the first column, and you get the second column of the matrix. And so on and so forth, you can figure out each column of the matrix. Finally, the Nth column is printed.

Computational complexity of algorithm:

F(N) = N + NlogN + N = 2N + NlogN

T(N) = O(NlogN)

Source code:

import java.io.\*;

import java.util.\*;

public class App {

static class NC {

char ch;

int next;

NC(char ch, int next) {

this.ch = ch;

this.next = next;

}

}

public static void main(String[] args) throws Exception {

StreamTokenizer in = new StreamTokenizer(new BufferedReader(new InputStreamReader(System.in)));

PrintWriter out = new PrintWriter(new OutputStreamWriter(System.out));

in.wordChars('\_', '\_');

in.nextToken();

int N = (int) in.nval;

in.nextToken();

String input = in.sval;

int n = input.length();

NC[] node = new NC[n];

char[] str = new char[n];

for (int i = 0; i < n; i++) {

char temp = input.charAt(i);

str[i] = temp;

node[i] = new NC(temp, i);

}

Arrays.sort(node, new Comparator<NC>() {

@Override

public int compare(NC o1, NC o2) {

boolean flag = o1.ch < o2.ch || o1.ch == o2.ch && o1.next < o2.next;

int result;

if (flag) {

result = -1;

return result;

} else {

result = 1;

return result;

}

}

});

int x = N - 1;

for (int i = 0; i < n; i++) {

x = node[x].next;

out.print(str[x]);

}

out.flush();

}

}