Strings

Representação de strings: problemas resolvidos

Prof. Edson Alves

2018

Faculdade UnB Gama

Sumário

Codeforces Round #163 – Problem A: Stones on the Table

Problema

There are n stones on the table in a row, each of them can be red, green or blue. Count the minimum number of stones to take from the table so that any two neighboring stones had different colors. Stones in a row are considered neighboring if there are no other stones between them.

Entrada e saída

Input

The first line contains integer n $(1 \leq n \leq 50)$ – the number of stones on the table.

The next line contains string s, which represents the colors of the stones. We'll consider the stones in the row numbered from 1 to n from left to right. Then the i-th character s equals "R", if the i-th stone is red, "G", if it's green and "B", if it's blue.

Output

Print a single integer – the answer to the problem.

Exemplo de entradas e saídas

Sample Input	Sample Output
3 RRG	1
5 RRRRR	4
4 BRBG	0

Solução com complexidade $O(n)$

- O problema consiste em determinar o número de remoções a serem realizadas de modo que caracteres vizinhos sejam distintos
- Para tal, basta observar todos os caracteres de s em sequência, um por vez, e manter o registro do último caractere que foi observado
- Este registro pode ser inicializado com um valor sentinela que não pode ocorrer na string (por exemplo, o caractere espaço em branco)
- Caso o caractere a ser observado é diferente do anterior, basta atualizar o anterior com o atual e prosseguir
- Caso seja idêntico ao anterior, é necessário removê-lo
- Neste caso não é necessário atualizar o valor do anterior
- Esta solução tem complexidade $O(n)$, pois visita cada caractere uma única vez

Solução AC com complexidade $O(n)$

```cpp
#include <bits/stdc++.h>

using namespace std;

int solve(const string& s)
{
    int ans = 0;
    char last = ' ';

    for (const auto& c : s)
    {
        if (c == last)
            ++ans;
        else
            last = c;
    }

    return ans;
}

```

Solução AC com complexidade $O(n)$

```
21 int main()
22 {
23     ios::sync_with_stdio(false);
24
25     int n;
26     string s;
27
28     cin >> n >> s;
29
30     auto ans = solve(s);
31
32     cout << ans << '\n';
33
34     return 0;
35 }
```

AtCoder Beginner Contest 109 – Problem B: Shiritori

Problema

Takahashi is practicing *shiritori* alone again today.

Shiritori is a game as follows:

- In the first turn, a player announces any one word.
- In the subsequent turns, a player announces a word that satisfies the following conditions:
  - That word is not announced before.
  - The first character of that word is the same as the last character of the last word announced.

In this game, he is practicing to announce as many words as possible in ten seconds.

You are given the number of words Takahashi announced, N, and the i-th word he announced, W_i, for each i. Determine if the rules of shiritori was observed, that is, every word announced by him satisfied the conditions.

Entrada e saída

Constraints

- N is an integer satisfying $2 \leq N \leq 100$.
- $W_i$is a string of length between 1 and 10 (inclusive) consisting of lowercase English letters.

Input

Input is given from Standard Input in the following format:

$$
\begin{aligned}
&N \\
&W_1 \\
&W_2 \\
&\vdots \\
&W_N
\end{aligned}
$$

Entrada e saída

Output

If every word announced by Takahashi satisfied the conditions, print 'Yes'; otherwise, print 'No'.

Exemplo de entradas e saídas

Exemplo de Entrada	Exemplo de Saída
4 hoge english hoge enigma	No
9 basic c cpp php python nadesico ocaml lua assembly	Yes

Solução

- A solução do problema consiste em observar as duas regras básicas do jogo
- Para tal, seja s_1 a primeira palavra dita e c o último caractere de s
- Para manter a primeira regra, as palavras já ditas devem ser mantidas em um conjunto
- Logo s_1 deve ser inserida neste conjunto
- Para as demais palavras s_i, com $2 \leq i \leq N$, deve-se verificar primeiramente a segunda regra, isto é, checar se o primeiro caractere de s_i é igual a c
- Caso não seja, a resposta do problema é 'No'
- Em caso afirmativo, s_i deve ser inserida no conjunto: caso já esteja lá, a resposta do problema também é 'No'
- Se não estiver no conjunto, o valor de c deve ser atualizado para o último caractere de s_i e continuar a rotina
- Esta solução tem complexidade $O(N \log N)$, devido as inserções no conjunto

Solução AC com complexidade $O(N \log N)$

```cpp
#include <bits/stdc++.h>

bool solve(int N, const std::vector<std::string>& words)
{
    auto c = words[0].back();

    std::set<std::string> S;
    S.insert(words[0]);

    for (int i = 1; i < N; ++i)
    {
        if (S.count(words[i]) or words[i].front() != c)
            return false;

        c = words[i].back();
        S.insert(words[i]);
    }

    return true;
}

```

Solução AC com complexidade $O(N \log N)$

```
22 int main()
23 {
24     std::ios::sync_with_stdio(false);
25
26     int N;
27     std::cin >> N;
28
29     std::vector<std::string> words(N);
30
31     for (int i = 0; i < N; ++i)
32         std::cin >> words[i];
33
34     auto ans = solve(N, words);
35
36     std::cout << (ans ? "Yes\n" : "No\n");
37
38     return 0;
39 }
```

Referências

1. Codeforces Round #163 – Problem A: Stones on the Table
2. AtCoder Beginner Contest 109 – Problem B: Shiritori