C6F 字符串转转转

21376218-杨凯伟

2023/12/6

题目描述

Abyss 最近拿到了一个全部由小写字母组成的字符串 $s=s_1s_2s_3\cdots s_n$。有一天他突发奇想，将该字符串旋转了 i 个单位，变成了新的字符串 $s'=s_{i+1}s_{i+2}\cdots s_ns_1s_2\cdots s_i$。现在他想知道，对于一个长度为 n 的字符串，能否实现旋转特定单位之后形成的新字符串与给定字符串一致。具体来说，给定两个长度为 n 的字符串 s,t，请你求出能否实现对 s 旋转 $i(1\le i<n)$ 个单位后与字符串 t 一致。

输入

第一个数为数据组数 $T(1\le T\le 20)$接下来 2T 行，对于每组数据：

第一行 1 个字符串表示 $s(1<|s|\le 2\times 10^5)$

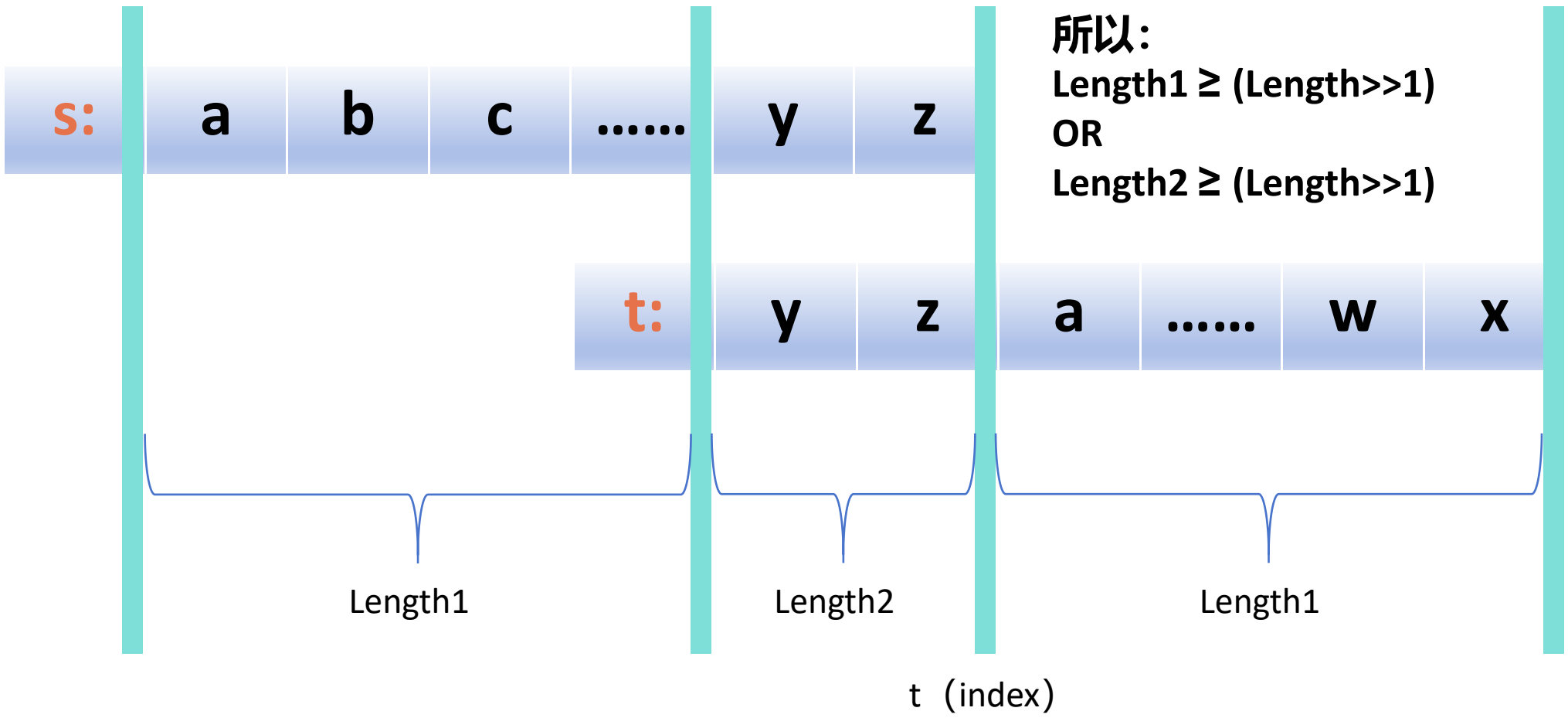
第二行 1 个字符串表示 $t(1<|t|\le 2\times 10^5)$，保证两字符串长度一致。

输出

对于每组数据，输出 T 行，如果存在特定旋转使得两字符串一致，则输出 Yes，否则输出 No。

一句话题意：判断一个字符串后缀等于另一个字符串前缀，同时剩余部分相等

解法一：KMP查重叠



因为：

Length1+Length2=Length

所以：

Length1 ≥ (Length>>1)

OR

Length2 ≥ (Length>>1)

1. 只需要判断 s 的前半段是否 t 的子串 或 t 的前半段是否是 s 的子串
2. 遍历字符串并检验

解法一：代码

```cpp
#include<bits/stdc++.h>
using namespace std;
const int N = 200005;
char s[N], t[N];
int nxt[N];
int kmp(int n, char *a, int m, char *b) {
    int i, j, ans = 0;
    for (nxt[0] = j = -1, i = 1; i < n; nxt[i++] = j) {
        while (~j && a[j + 1] != a[i]) j = nxt[j];
        if (a[j + 1] == a[i]) j++;
    }
    for (j = -1, i = 0; i < m; i++) {
        while (~j && a[j + 1] != b[i]) j = nxt[j];
        if (a[j + 1] == b[i])j++;
        if (j == n - 1) ans=i-n+1, j = nxt[j];
    }
    return ans;
}
```

```cpp
int main() {
    int m;cin >> m;
    while (m--) {
        bool flag=true; int index;
        scanf("%s", s); scanf("%s", t);
        int len = strlen(t);
        if(len==1&&s[0]==t[0]){ cout<<"Yes"<<endl; continue; }
        if((index=kmp(len/2, t, len, s))){//s后缀与t前缀
            for(int i=0;i<index;i++)//遍历s前缀
                if(s[i]!=t[len-index+i])
                    flag=false;
            for(int i=0;i<len-index;i++)//遍历s后缀
                if(s[index+i]!=t[i])
                    flag= false;
        }else if((index=kmp(len/2,s,len,t))){//t后缀与s前缀
            for(int i=0;i<index;i++)
                if(t[i]!=s[len-index+i])//遍历t前缀
                    flag=false;
            for(int i=0;i<len-index;i++)//遍历t后缀
                if(t[index+i]!=s[i])
                    flag= false;
        }else//如果不存在
            flag= false;
        if(flag) cout<<"Yes"<<endl;
        else cout<<"No"<<endl;
    }
}
```

解法二：字符串拼接

只需要判断 t 是 S’ 的子串即可

注意：需要考虑到**字符串相等**的情况：

若s=t=”aaa”，应当输出”Yes\n”;
若s=t=’’asd”，应当输出”No\n”。

结果	得分	语言	代码长度 (Bytes)	运行时间 (ms)	运行内存 (KB)
Wrong Answer	0	c++	296	87	4136
Wrong Answer	0.75	c++	386	97	4160
	75	c++	388	100	4016
	75	c++	368	97	4136
	75	c++	355	99	4084
	0.75	c++	283	630	3940
Wrong Answer	0	c++	220	23	3592
Wrong Answer	0.75	c++	266	93	4192
Wrong Answer	0.75	c++	368	910	4236
Wrong Answer	0	c++	357	659	4204
Wrong Answer	0.75	c++	355	94	4144
Wrong Answer	0.75	c++	443	93	4136
Wrong Answer	0.75	c++	261	93	4104
Wrong Answer	0.75	c++	274	314	4108
Wrong Answer	0.75	c++	306	847	3944

Wrong Answer | all

Accepted | 1 * (1 / 4) | 30 ms | 4160 KB
Wrong Answer | 0 * (1 / 4) | 2 ms | 3512 KB Accepted | 1 * (1 / 4) | 30 ms | 4108 KB Accepted | 1 * (1 / 4) | 35 ms | 4064 KB

解法二：代码展示

```c
//C_version
#include <stdio.h>
#include <string.h>
char s[400009], t[200009];
int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        scanf("%s%s", s, t);
        sprintf(s, "%s%s", s, s);s[strlen(t)*2-1]='\0';
        printf(strstr(s+1, t) ? "Yes\n" : "No\n");
    }
}
```

```cpp
//C++_version
#include <bits/stdc++.h>
using namespace std;
int main() {
    string s, t;int T;cin>>T;
    while(T--) {
        cin >> s >> t;
        s=s.append(s.substr(0,s.length()-1)).substr(1);
        cout << (s.find(t) < t.length() ? "Yes\n" : "No\n");
    }
}
```

解法三：标程

```cpp
#include <bits/stdc++.h>
using namespace std;
const int MAXN = 1e6 + 5;
struct Hash {
    int BASE, P, val[MAXN], pw[MAXN];
    void init(int base, int p, string s) {
        BASE = base, P = p;
        int n = s.size();
        val[0] = s[0];
        for (int i = 1; i < n; i++)
            val[i] = ((long long) val[i - 1] * base + s[i]) % p;
        pw[0] = 1;
        for (int i = 1; i <= n; i++) pw[i] = (long long) pw[i - 1]
    }
    int query(int l, int r) {
        if (l) return (val[r] + P - (long long) val[l - 1] * pw[r
        return val[r];
    }
};
```

```cpp
Hash hs, ht;
string s, t;
int main() {
    int T;
    for (cin >> T; T--;) {
        cin >> s >> t;
        s = s + s;
        hs.init(131, 1000000007, s);
        ht.init(131, 1000000007, t);
        int len = t.size();
        bool got = false;
        for (int i = 1; i < len; i++)
            if (hs.query(i, i + len - 1) == ht.query(0, len - 1)) {
                got = true;
                break;
            }
        if (got) cout << "Yes\n";
        else cout << "No\n";
    }
    return 0;
}
```

性能比较

	代码长度(Bytes)	运行时间(ms)	运行内存(KB)
解法一 C++	1532	203	4272
解法二 C_version	283	92	2288
解法二 C++_version	262	629	4392
标程 C++	1141	1459	8992