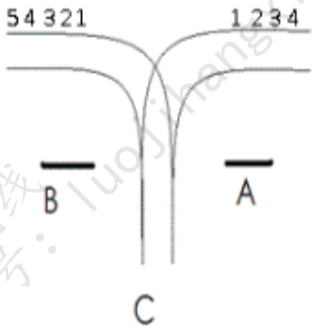
1357：车厢调度(train)

题目描述

有一个火车站，铁路如图所示，每辆火车从A驶入，再从B方向驶出，同时它的车厢可以重新组合。假设从A方向驶来的火车有n节（n≤1000），分别按照顺序编号为1，2，3，…，n。假定在进入车站前，每节车厢之间都不是连着的，并且它们可以自行移动到B处的铁轨上。另外假定车站C可以停放任意多节车厢。但是一旦进入车站C，它就不能再回到A方向的铁轨上了，并且一旦当它进入B方向的铁轨，它就不能再回到车站C。



负责车厢调度的工作人员需要知道能否使它以a1,a2,…,an的顺序从B方向驶出，请来判断能否得到指定的车厢顺序。

输入格式

第一行为一个整数n，其中n≤1000，表示有n节车厢，第二行为n个数字，表示指定的车厢顺序。

输出格式

如果可以得到指定的车厢顺序，则输出一个字符串“YES”，否则输出“NO”（注意要大写，不包含引号）。

输入样例

```
5
5 4 3 2 1
```

输出样例

```
YES
```

解析

依然是一道阅读理解题，首先这道题绝对不是让你比较两个序列颠倒之后是否相同！

对于入栈序列，它可以有多种多样的出栈顺序，而我们就是要在它众多的顺序中判断序列2是否使其中之一。

举例，对于栈1 2 3 4 5，它的出栈顺序可能有：

a、1 2 3 4 5，即入1个马上就出一个。

b、2、1、3、4、5，即入到2，然后全部出栈，接着入1个出1个。

题意搞懂了，那么怎么判断出栈序列是否是入栈序列众多出栈方式之一呢？很简单，我们将序列1进行入栈，当它的栈顶与出栈序列相同时，就执行出栈。如果入栈序列最后能够全部出栈，则表示两个序列是匹配的。

编码

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

using namespace std;
stack<int> q;//栈q
//入栈队列a，待检验队列b
vector<int> a, b;
int p, n;//p组数据，n为序列长度
int main() {
    cin >> n;
    //计数器sum
    int sum = 0;
    //读入数据
    for (int i = 1; i <= n; i++) {
        a.push_back(i);
    }
    //存储目标出栈顺序
    for (int i = 0; i < n; i++) {
        int t;
        cin >> t;
        b.push_back(t);
    }
    //尝试是否存在指定的出栈顺序
    for (int i = 0; i < n; i++) {
        q.push(a[i]);//入栈
       //当栈顶元素与b中当前元素相同时出栈
       while (!q.empty() && (q.top()) == b[sum]) {
            //尝试出栈
          q.pop();
            //sum++到b下一个元素
          sum++;
```

```
            }
        }
        //如果栈为空说明出栈序列b正确
        if (q.empty()) {
            cout << "YES" << endl;
        } else {
            cout << "NO" << endl;
        }
        return 0;
}
```