**Number Sequence**

**Time Limit: 10000/5000 MS (Java/Others)    Memory Limit: 32768/32768 K (Java/Others)  
Total Submission(s): 19450    Accepted Submission(s): 8352**

Problem Description

Given two sequences of numbers : a[1], a[2], ...... , a[N], and b[1], b[2], ...... , b[M] (1 <= M <= 10000, 1 <= N <= 1000000). Your task is to find a number K which make a[K] = b[1], a[K + 1] = b[2], ...... , a[K + M - 1] = b[M]. If there are more than one K exist, output the smallest one.

Input

The first line of input is a number T which indicate the number of cases. Each case contains three lines. The first line is two numbers N and M (1 <= M <= 10000, 1 <= N <= 1000000). The second line contains N integers which indicate a[1], a[2], ...... , a[N]. The third line contains M integers which indicate b[1], b[2], ...... , b[M]. All integers are in the range of [-1000000, 1000000].

Output

For each test case, you should output one line which only contain K described above. If no such K exists, output -1 instead.

Sample Input

2

13 5

1 2 1 2 3 1 2 3 1 3 2 1 2

1 2 3 1 3

13 5

1 2 1 2 3 1 2 3 1 3 2 1 2

1 2 3 2 1

Sample Output

6

-1

第二个字符串是否可以在第一个字符串出现 找到的话 输出第一个字符的位置 没找到 就输出-1

直接套用KMP模板：

#include <stdio.h>

#include <string.h>

int a[1000005],b[10005];

int next[10005];

int n,m;

void Next() ///求next数组

{

next[0] = next[1] = 0;

for(int i = 1; i < m; i++)

{

int j = next[i];

while(j&&b[j]!=b[i])

j = next[j];

next[i+1] = b[i]==b[j]?j+1:0;

}

}

void find1() ///模式串与主串进行匹配

{

int j = 0; ///初始化在模式串的第一个位置

for(int i = 0; i < n; i++) ///遍历主串

{

while(j&&b[j]!=a[i])

j = next[j];

if(b[j]==a[i]) ///如果匹配成功则进行下一个位置

j++;

if(j==m) ///如果能在a数组中找到b数组 则直接输出

{

printf("%d\n",i-m+2);

return ;

}

}

printf("-1\n");

}

int main()

{

int t;

scanf("%d",&t);

while(t--)

{

scanf("%d%d",&n,&m);

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

scanf("%d",&a[i]);

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

scanf("%d",&b[i]);

Next();

find1();

}

return 0;

}