ACGT

Input file:	`stdin`
Output file:	`stdout`
Time limit:	4 seconds
Memory limit:	512 megabytes

Usámec is trying to construct a genome from DNA fragments. A DNA fragment is a sequence of letters $[ACGT]$. Usámec has found N fragments already, let's denote them as $F_{1..N}$. There are also M rules describing which fragments can follow directly after which ones.

Constructing a DNA

DNA is a sequence of fragments F_{a_i} (for some sequence $\{a_i\}_{i=1}^{k}$; $1 \leq a_i \leq N$) such that for every $i > 1$, the a_i-th fragment can follow after the a_{i-1}-st one. We construct the DNA by concatenating those fragments and obtain a string of letters $[ACGT]$ as a result.

Distance of strings

We can modify a string in steps, where each step is one of these 3 types of changes: adding a character into the string, removing or changing one character in the string. We define the *distance* of strings p and q as the smallest number of changes we need to make in p to turn it into q (clearly, any string has distance 0 from itself). For example, $ATTA$ and AGA have distance 2 — we need to remove one T and change the other into G.

You're given some DNA fragments and rules that describe which fragment can follow after which one. You're also given a target DNA string r, a *very small* non-negative integer d and indices of 2 fragments, u and v ($u \neq v$). You should write a program that finds *one possible* sequence of fragments that starts with fragment u, ends with fragment v and the DNA that's constructed by it has distance from r of at most d. Specifically, it's not required to minimize this distance.

You may assume that no fragment can follow directly after itself and that if fragments y and z can follow after x, then the first characters of F_y and F_z are different.

Input

The first line of the input contains the number of fragments N and number of rules M. The following N lines contain one fragment F_i each. The following M lines contain 2 integers a, b (1 rule) each — it means that fragment F_b can follow directly after fragment F_a. The next-to-last line contains integers d, u, v and the last one contains string r.

Output

In case there's no sequence of fragments that satisfies the conditions, print one line with one integer -1. Otherwise, print 2 lines - the size of your sequence on the first line and this sequence, as space-separated indices of fragments, on the second one.

Examples

stdin	stdout
3 3 CG AT TCC 1 2 2 1 1 3 0 1 3 CGATCGTCC	1 2 1 3
4 4 CCC A T GGG 1 2 2 3 3 2 2 4 1 1 4 CCCATTGGG	1 2 3 2 4
4 4 CCC A T GGG 1 2 2 3 3 2 2 4 1 1 4 CCCAAAAGGG	-1

Note

The input files are based on real data, which means there won't be many similar fragments.

There are 10 tests; in all of them, $d \leq 5$, $n \leq 1000$, $m \leq 4000$ and both the total length L_T of all fragments and R (the length of r) won't exceed 10^6. In addition, the following restrictions hold:

Set(s) no.	Restriction
1-8	$L_T \leq 5 \cdot 10^4, R \leq 5 \cdot 10^4$
2-3, 7-8	$L_T \leq 1000, R \leq 1000$
1	$d = 0$
6	$d = 1$
2-5	$n = 2, m = 1$
9-10	$m \leq 1000$