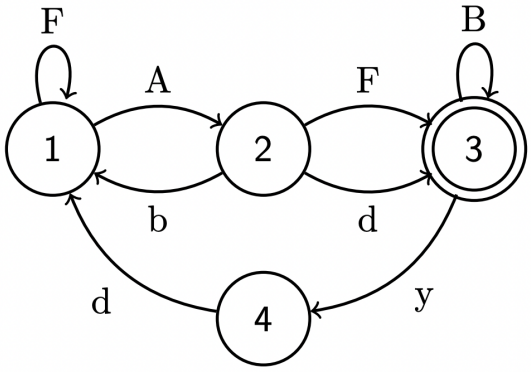
Problem K
Transparency
Time Limit: 6 Second(s)

You must audit a factory that produces a number of products. Each product is subject to its own taxation rules and regulations. Unfortunately, deciding which rules apply to a product is not obvious since it can be difficult to distinguish the products from one another. It is your job to investigate if the factory is being completely *transparent* in its production process by determining if the factory is capable of producing two different products that are indistinguishable from one another.



The factory design from the first sample input. The packaging station is marked by a double circle.

The final products are represented by sequences of uppercase letters ('A'–'Z') and lowercase letters ('a'–'z'). Two final products are indistinguishable from one another if the products are equal after removing all lowercase letters. For example, `WabXYcz` is indistinguishable from `gWXfbY`, since after removing all lowercase letters we are left with `WXY` in both instances.

A factory is modelled as a set of stations S, a set of directed and labelled transition edges T, an initial station s_0, and a set of packaging stations P. The initial station is in the set of stations, i.e., $s_0 \in S$, and the set of packaging stations is a subset of S, i.e., $P \subseteq S$. A transition edge is defined by a 3-tuple (s_1, σ, s_2), where $s_1, s_2 \in S$ are stations, and σ is an uppercase or lowercase letter. The existence of the transition (s_1, σ, s_2) in T means that if in producing some product we are at station s_1, then appending σ to the product will leave us at station s_2. That is, there is a directed edge from station s_1 to station s_2, labelled σ.

A product can be produced by the factory if by starting at station s_0 there is a sequence of edges that can be followed, ending at a station in P, whose edge labels can be concatenated, in order, to create the product. The sequence of edges can be empty if $s_0 \in P$. For example, the following strings can all be produced by the factory described in the first sample input and illustrated in the figure: `AF`, `FAFB`, `AbFFAd`, `AdydAd`. Note that this is not an exhaustive list.

Each production transition can be used an unlimited number of times to make a product. It is guaranteed that for each station, s, and letter, σ, there is at most one directed edge from station s labelled σ. That is, $(s_1, \sigma, s_2), (s_1, \sigma, s_3) \in T$ implies $s_2 = s_3$. It is possible that transitions will go back to the same station; that is, $(s_1, \sigma, s_1) \in T$ is allowed.

Given a factory design, determine if the factory is capable of producing two distinct but indistinguishable products. If there is such a pair of products, then report the sum of the lengths of the strings in the pair that minimizes the sum of the lengths. If there is no such pair, print -1.

Input

The first line of input contains three integers, s ($1 \leq s \leq 50$), p ($1 \leq p \leq s$) and t ($1 \leq t \leq 52 \cdot s$), where s is the number of stations, p is the number of packaging stations, and t is the number of transitions. The set of stations is numbered from 1 to s.

Each of the next p lines contains a single integer i ($1 \leq i \leq s$). These are the packing stations. All values of i are distinct.

Each of the next t lines is of the form:

s_1 σ s_2

where s_1 and s_2 ($1 \leq s_1, s_2 \leq s$) are stations, and σ is a single character, consisting of an uppercase or lowercase letter. These are the transitions. The initial station is always station 1. There will never be two transitions out of the same state labelled with the same character.

Output

Output a single line with a single integer. If there is no pair of distinct, indistinguishable products, then output -1. If there exists a pair of distinct, indistinguishable products, then output the smallest sum $|a| + |b|$ where a and b are strings corresponding to distinct, indistinguishable products.

Sample Input 1	Sample Output 1
4 1 8 3 1 F 1 1 A 2 2 b 1 2 F 3 2 d 3 3 B 3 3 y 4 4 d 1	10

Sample Input 2	Sample Output 2
4 1 8 3 1 F 1 1 A 2 2 b 1 2 F 3 2 d 3 3 B 3 3 y 4 4 d 4	-1

Sample Input 3	Sample Output 3
5 2 5 1 5 1 i 2 2 c 3 3 p 4 4 c 1 1 I 5	4