Modernise

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

Presidente, congratulations on your election as the new leader of the beautiful island of Tropico! Unfortunately, your predecessor has left the island in shambles, and it is up to you to modernise many aspects of our society.

The first item on your agenda is to give the island a first-rate road system. Until now, the n towns on the island have been linked by a network of m dirt roads, with every town able to reach every other town by following at least one road. Unfortunately, our policy of importing cars from all over the world has resulted in a chaotic mix of left-hand driven and right-hand driven cars throughout the island.

Your loyal advisor Penultimo has suggested that you instead separate the road network. Some towns should allow only left-hand driven cars, and they will be connected by roads where one drives on the right. The remaining towns will be the opposite, with only right-hand driven cars, on the left side of the road. Your adoring citizens will gladly reorganise their entire lives to suit your arbitrary decrees, so you have complete freedom to choose which towns will allow which type of cars.

You will then have to perform major maintenance on the roads between these towns. However, our island is still very poor, and every mile of road that is to be sealed, have signs installed, etc., will be very expensive. Thus you have decided to only maintain the smallest possible length of road to ensure that every left-hand driven town can reach every other, and that every right-hand driven town can reach every other. What is the minimum length of road we need to maintain?

Input

The first line of input will contain a two integers n and m ($2 \leq n \leq 100,000$), the number of towns and the number of existing roads.

m lines follow, the i-th line containing three integers a_i, b_i and ℓ_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$, $0 \leq \ell_i \leq 10,000$), representing that the i-th road joins towns a_i and b_i and has length ℓ_i.

Output

Output one line, the minimum length of road that must be maintained so that all left-hand driven towns are accessible from all others and likewise for right-hand driven towns.

Scoring

For Subtask 1 (50 points):

- $m = n - 1$.

For Subtask 2 (50 points):

- $n - 1 \leq m \leq 100,000$.

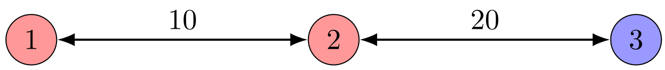
Examples

standard input	standard output
3 2 1 2 10 2 3 20	10
3 3 1 2 30 1 3 20 2 3 40	20

Note

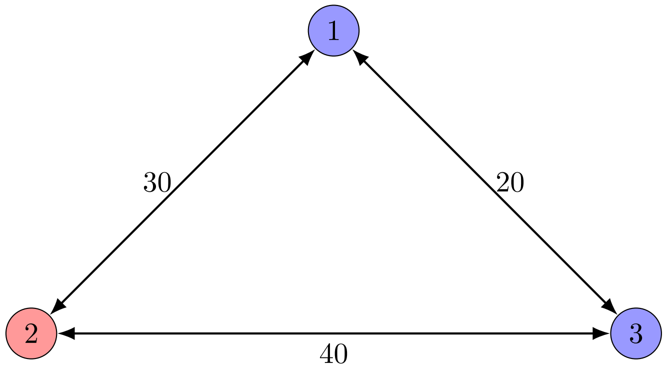
In the first sample case, there are two roads between three towns.

We can designate the first two towns as left-hand driven and the third as right-hand driven, so only the first road (of length 10) needs maintenance.



In the second sample case, there are three roads between three towns. *Note that this does not comply with the constraints of Subtask 1.*

We can designate the first and third towns as right-hand driven and the second as left-hand driven, so only the second road (of length 20) needs maintenance.



Avatar Tour

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The Avatar continent was once a harmonious place, featuring a very mountainous terrain with n mountains, each with some height h_i, connected by m bidirectional roads that connect pairs of mountains.

Ever since the great war, the continent has been split into k nations. As a result, each of the n mountains now belongs to exactly one of the k nations.

Denyce, an excited tourist, wishes to explore the Avatar continent. As part of her trip, she bought an electric car that recharges at a rate of c when going from higher mountains to lower mountains, and discharges energy at a rate of d when moving from lower mountains to higher mountains. More formally, when travelling along some road from a mountain of height h_i to another mountain of height h_j:

- If $h_i \geq h_j$, then the car gains $c(h_i - h_j)$ energy,
- If $h_i < h_j$, then the car loses $d(h_j - h_i)$ energy.

It is guaranteed that $d \geq c$.

Denyce wishes to visit at least one mountain of every nation using her electric car. She wants to do so by starting at any mountain and ending at any mountain, following some sequence of roads. A road may be used more than once.

Your task is to help Denyce find out what is the minimum amount of energy required to complete such a task, where energy required is starting energy minus final energy.

You may assume that the car has arbitrarily high energy initially, so it never runs out of energy during a climb.

Input

The first line contains three space-separated integers n, m and k ($1 \leq n \leq 50$, $0 \leq m \leq 200$), representing the number of mountains, roads and nations respectively.

The second line contains two space-separated integers c and d ($1 \leq c \leq d \leq 100$), representing the charge and discharge rate of the electric car respectively.

The next line of input consists of n space-separated integers $h_1, ..., h_n$ ($0 \leq h_i \leq 1,000$), the ith of which represents the height of the ith mountain.

The next line of input consists of n space-separated integers $p_1, ..., p_n$ ($1 \leq p_i \leq k$), the ith of which represents the nation that the ith mountain belongs to.

m lines follow, each describing one road. The jth such line consists of two space-separated integers a_j and b_j ($1 \leq a_j < b_j \leq n$), representing the two mountains connected by the jth road.

Output

Output a single integer, the minimum amount of energy required to visit at least one mountain of every nation.

If it is not possible to visit every nation at least once, output "`impossible`" without the quotes.

Scoring

For Subtask 1 (50 points):

- $k = 2$.

For Subtask 2 (50 points):

- $2 \le k \le 10$.

Examples

standard input	standard output
3 3 2 5 10 3 4 5 1 1 2 1 2 2 3 1 3	-10
6 6 3 6 9 10 3 10 6 4 10 1 2 1 2 2 3 1 2 2 3 2 4 4 5 4 6 5 6	-21
1 0 2 3 3 5 2	impossible

Note

In the first sample case, our path is $3 \to 1$.

Since this is a drop of height 2, we gain 10 energy and have a final cost of -10.

In the second sample case, our path is $6 \to 4 \to 2 \to 3 \to 2$.

Along the path we spend energy -24, -18, $+63$, -42, for a net expenditure of -21.

In the third sample case, there is no mountain in nation 1, so it's impossible to reach every nation.

Hippo Ponds

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In the Serengeti, there are m ponds in a row labelled 1 to m, and n baby hippos labelled 1 to n. Each baby hippo would like to choose a pond, subject to two conditions.

1. They each prefer certain ponds over others — if baby hippo i is in pond j, they will gain a positive amount of happiness $h_{i,j}$.
2. They prefer to be close to their friends — if two baby hippos are friends and they are d ponds apart from each other, each of these two loses $\frac{c}{2} \cdot d$ happiness (i.e. there is $c \cdot d$ happiness lost between them).

Now, it is your job to assign each baby hippo to a pond such that the sum of their happiness is maximised. Note that many baby hippos can share the same pond (the baby hippos are quite small, after all).

Input

The first line contains four integers n, m, k and c ($1 \leq n, m \leq 40$, $0 \leq k \leq 50$, $0 \leq c \leq 10^6$), representing the number of baby hippos, the number of ponds, the number of friendships and the penalty multiplier for friends being far away.

Each of the next n lines contains m space-separated integers. The j-th of these is $h_{i,j}$ ($0 \leq h_{i,j} \leq 10^6$), representing how happy the ith baby hippo is with the jth pond.

Each of the next k lines contains 2 integers u_i and v_i ($1 \leq u_i < v_i \leq n$), indicating that baby hippos u_i and v_i are friends. No friendship will appear twice in the input.

Output

Output an integer, the maximum sum of their happiness which can be achieved.

Scoring

For Subtask 1 (50 points):

- all u_i and v_i are distinct, i.e. each hippo has at most one friend.

For Subtask 2 (50 points):

- no additional constraints apply (beyond those listed in **Input** above).

Examples

standard input	standard output
3 3 1 5 2 4 10 10 1 12 9 7 1 1 3	24
3 3 2 2 10 5 7 4 2 6 3 6 3 1 2 2 3	18

Note

In the first sample case, baby hippos 1 and 3 are friends.

We can assign baby hippo 1 to pond 3, baby hippo 2 to pond 3 and baby hippo 3 to pond 2.

- Before considering friendships, the total happiness is $h_{1,3} + h_{2,3} + h_{3,2} = 10 + 12 + 7 = 29$.
- The friendship between baby hippos 1 and 3 incurs a total penalty of $C \cdot d = 5 \cdot 1 = 5$, since they are in neighbouring ponds.

The final happiness is $29 - 5 = 24$. This is the maximum obtainable happiness.

In the second sample case, baby hippos 1 and 2 are friends, and baby hippos 2 and 3 are friends. *Note that this does not comply with the constraints of Subtask 1.*

We can assign baby hippo 1 to pond 1, baby hippo 2 to pond 1 and baby hippo 3 to pond 2.

- Before considering friendships, the total happiness is $h_{1,1} + h_{2,1} + h_{3,2} = 10 + 4 + 6 = 20$.
- The friendship between baby hippos 1 and 2 incurs a total penalty of 0 since they are in the same pond.
- The friendship between baby hippos 2 and 3 incurs a total penalty of $C \cdot d = 2 \cdot 1 = 2$, since they are in neighbouring ponds.

The final happiness is $20 - 0 - 2 = 18$. This is the maximum obtainable happiness.