Problem G. Grachtenrace

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

The Netherlands is famous for its grachten, the city canals. To have some fun with the grachten, you decide to organize a grachtenrace with a group of friends. The start and finish locations are chosen, and each participant must try to go from start to finish by boat in the least amount of time possible.



Figure 7: Two speedboats chasing each other on the canals in Amsterdam.

The network consists of n junctions numbered 1 through n, with junction 1 corresponding to the start and junction n corresponding to the finish. There are m grachten, each connecting a unique unordered pair of junctions and having a given length in meters, which can be traversed in both directions. It is guaranteed that any junction can be reached from any other junction. One might assume there must exist a planar embedding of the network of grachten; this cannot be assumed due to the possibility of traversable sewers and aqueducts at heights different from the rest of the grachten.

There are docks at k junctions of the network: exactly one boat is stored at each dock. To make the race more interesting, the participants can switch boats at all docks. Each boat has a speed given in **seconds per meter**. Traveling with a boat of speed s along a gracht of length w will take $s \cdot w$ seconds. It is guaranteed that there is a dock at junction 1.

You have planned an exciting race and you are almost done with the preparations. Unfortunately, there has been an error in placing the boats: they have been mixed up completely, and as a result, the assignment of boats to docks is uniformly random. It is too late to fix this, so you decide to keep the race as it is. Determine the least amount of time in seconds possible to finish the race.

To recap: the junctions, the grachten, the docks, and the collection of boats and their speeds are all predetermined and not random, but it is guaranteed for each hidden test that the assignment of which boat is placed at which dock is picked uniformly at random. So, to emphasize, **only the assignment of the boats to the docks is random**.

Input

The first line of input contains three space-separated integers n, m, and k ($2 \le n \le 10^5$, $n-1 \le m \le 10^5$, $1 \le k \le n$), representing the number of junctions, the number of grachten, and the number of docks, respectively.

Each of the next k lines contains two space-separated integers b_i and s_i ($1 \le b_i \le n$, $1 \le s_i \le 10^6$), meaning that there is a dock at junction b_i containing a boat with speed of s_i seconds per meter. Each value of b_i is guaranteed to be unique, and exactly one value b_i equals 1.

Each of the next m lines contains three space-separated integers u_i, v_i, and w_i ($1 \le u_i < v_i \le n$, $1 \le w_i \le 10^6$), representing a gracht between junctions u_i and v_i with a length of w_i meters. Each pair (u_i, v_i) is guaranteed to be unique, and it is guaranteed that any junction can be reached from any other junction.

Additionally, it is guaranteed that, at the end of the creation of each hidden test, the values s_i were randomly permuted.

Output

Output a single integer denoting the least amount of time in seconds possible to race from junction 1 to junction n.

Examples

standard input	*standard output*
4 4 2 1 50 2 10 1 2 5 2 3 10 3 4 5 1 4 15	400
4 4 1 1 50 1 2 5 2 3 10 3 4 5 1 4 15	750

Note

In the first example, there are 2 docks at junctions 1 and 2 containing boats with speeds 50 and 10 respectively. You can first travel from junction 1 to junction 2 using the first boat, which takes $50 \cdot 5 = 250$ seconds. Then you can switch to the second boat and move via junction 3 to junction 4, which takes an additional $10 \cdot (10 + 5) = 150$ seconds, so this path will take 400 seconds in total. It can be proven that this is optimal.

Note that the path from junction 1 to junction 4 is shorter (15 meters vs 25 meters), but because it must be traversed using the slower boat with speed 50, this takes $50 \cdot 15 = 750 > 400$ seconds.