SEPARATION

Everything is perfect. News articles call you the next unicorn, your employee count doubles on a biweekly basis, and you sleep peacefully knowing that the Enemy-o-Matic has recently been used to harm its 100,000th friendship.

Out of nowhere, you discover that the wife of your CTO (the technical cofounder who had the high GPA and keeps everything running) got matched to a residency in Oakland, and so he's decided to leave the company and accept a professorship at UC Berkeley. This would spell disaster for the Enemy-o-Matic, so to prevent his departure you aim to ensure he cannot make the trip to California.

The country can be modeled as having N airports, where airport 1 is BOS, airport 2 is OAK, airport 3 is SFO, and airport 4 is SJC. For the i'th of M pairs of airports, a_i, b_i, you know there are c_i flight tickets remaining from a_i to b_i, each with cost d_i. Additionally, if the situation gets very desperate, you can even bribe the TSA to lock down some airports. This costs e_j for airport j.

You know your cofounder has not yet purchased his tickets. He's willing to pay any amount for the trip, but if no itinerary exists between BOS and one of {OAK, SFO, SJC}, he and his wife (although she already has tickets) will reluctantly stay. The fate of the Enemy-o-Matic rests on you being able to ensure he stays with the company, so don't let everyone down!

Input

The first line contains two space-separated integers, N and M.

The next M lines list the available flight tickets. The i'th of these contains four space-separated integers a_i, b_i, c_i, d_i indicating that c_i tickets costing d_i dollars each remain from airport a_i to airport b_i. It is guaranteed that $1 \leq a_i, b_i \leq N, a_i \neq b_i$, that all ordered pairs (a_i, b_i) will be distinct, and that $1 \leq c_i, d_i \leq 10^9$.

The final line contains N space-separated integers, $e_1, \cdots, e_N$, where $1 \leq e_1, \cdots, e_N \leq 10^9$.

Output

Output the minimum amount of money needed to guarantee your CTO stays. This means that you can purchase enough flight tickets and/or cause enough airports to go into lockdown such that no possible trip remains between BOS (airport 1) and the Bay Area (airports 2, 3, 4).

In particular, you must ensure it is not the case that there is a sequence of airports $l_1, \cdots, l_\ell$ such that $l_1 = 1, l_\ell \in \{2, 3, 4\}$, none of the l_k are locked down, and at least one ticket remains between l_k and l_{k+1} for all $1 \leq k < \ell$.

Constraints

Let L be the maximum of all c_i and d_i. In all test cases, $4 \leq N \leq 60$ and $1 \leq M \leq N(N-1)$. Beyond the sample input, the tests are divided into batches with additional constraints. Time limits below are for C/C++; Ocaml gets 2x, Java 3x, and Python 10x.

- 24 points satisfy $N = 4$. TL: 100ms.
- 27 points satisfy $e_j = 10^9$ for all j and $N \cdot L^2 \leq 10^5$. TL: 500ms.

- 33 points satisfy $N \cdot L^2 \leq 10^5$. TL: 500ms.
- 16 points satisfy no further constraints. TL: 500ms.

Sample explanation

One optimal solution involves buying the ticket between airports 1 and 2 and the ticket between airports 5 and 4, and bribing the TSA to lock down airport 3. This has a total cost of $10 + 1 + 9 = 20$.

View submissions (https://cs124.seas.harvard.edu/problem/SEPARATION/code-submission)

Test cases

Input	Output	Points	Timeout

`5 4`
`1 2 1 10`
`1 3 2 7`
`1 5 1 2`
	`20`	0	100 ms

Hidden	*Hidden*	8	100 ms
Hidden	*Hidden*	8	100 ms
Hidden	*Hidden*	8	100 ms
Hidden	*Hidden*	9	500 ms
Hidden	*Hidden*	9	500 ms
Hidden	*Hidden*	9	500 ms
Hidden	*Hidden*	11	500 ms
Hidden	*Hidden*	11	500 ms
Hidden	*Hidden*	11	500 ms
Hidden	*Hidden*	8	500 ms
Hidden	*Hidden*	8	500 ms

Download (https://cs124.seas.harvard.edu/problem/SEPARATION/test-cases)

Inspired by the "Ultra Cool Programming Contest Control Centre" by Sonny Chan.
Modified for CS 124 by Neal Wu (https://github.com/nealwu), with design help from Martin Camacho.
Further refined by Nikhil Benesch (https://github.com/benesch).