CODEFORCES
Sponsored by TON

CSE221 Assignment 07 Summer 2025

A. Shortest Path

time limit per test: 1 second
memory limit per test: 1024 megabytes

You are given an **directed weighted** graph with N nodes and M edges. The nodes are numbered from 1 to N. The graph contains no self-loops or multiple edges.

There is a source and a destination. Your task is to find the shortest distance from the source node to the destination node and print the path taken. If multiple shortest paths exist, print any one of them. If no such path exists, print -1.

Input

The first line contains four integers $N, M, S, D(2 \le N \le 2\times10^5, 1 \le M \le 3\times10^5, 1 \le S, D \le N)$ — the number of vertices, total number of edges, source and destination.

The second line contains M integers $u_1, u_2, u_3 \ldots u_m$ $(1 \le u_i \le N)$ — where the i-th integer represents the node that is one endpoint of the i-th edge.

The third line contains M integers $v_1, v_2, v_3 \ldots v_m$ $(1 \le v_i \le N)$ — where the i-th integer represents the node that is the other endpoint of the i-th edge.

The fourth line contains M integers $w_1, w_2, w_3 \ldots w_m$ $(1 \le w_i \le 10^6)$ — where the i-th integer represents the weight of the i-th edge.

The i-th edge of this graph is from the i-th node in the second line to the i-th node in the third line.

Output

- If a valid path exists from S to D, print the shortest distance S to D on the first line.
- On the second line, print the nodes in the path in order from S to D. If multiple shortest paths exist, print any one of them.
- If no such path exists, print -1.

Examples

input
```
4 3 4 2
1 3 4
2 2 3
3 4 5
```
output
```
9
4 3 2
```

input
```
6 5 1 5
1 4 1 6 4
2 1 6 2 6
3 3 4 3 4
```
output
```
-1
```

input
```
2 1 2 1
2
1
7
```
output
```
7
2 1
```

input
```
5 7 2 4
1 1 5 4 2 3 2
5 4 3 5 5 4 3
3 8 2 6 6 4 3
```
output
```
7
2 3 4
```

B. Where to Meet?

time limit per test: 2 seconds
memory limit per test: 1024 megabytes

Alice and Bob are in a hurry to meet each other and have to traverse through a **directed** graph with **weighted** edges. The nodes are numbered from 1 to N. The graph contains no self-loops or multiple edges.

Alice starts from node S and Bob starts from node T. They want to find a common node in the graph where they can meet each other in the minimum amount of time. Alice or Bob can wait at any node if they want to.

Input

The first line contains four integers $N, M, S, T(2 \le N \le 2\times10^5, 1 \le M \le 3\times10^5, 1 \le S, T \le N)$ — the number of vertices, the total number of edges, the starting node of Alice, and the starting node of Bob.

The next M lines will contain three integers $u_i, v_i, w_i(1 \le u_i, v_i \le N, 1 \le w_i \le 10^6)$ — there is a edge from the node u_i to the node v_i with a weight w_i.

Output

Print two integers separated by a space: the minimum time required for Alice and Bob to meet, and the node where they meet. If there are multiple such nodes, print the smallest node. If no such node exists, print -1.

Examples

input
```
5 5 1 5
1 2 1
2 3 1
5 3 2
1 4 2
5 4 2
```
output
```
2 3
```

input
```
6 5 1 5
1 2 3
4 1 3
1 6 4
6 2 3
4 6 4
```
output
```
-1
```

input
```
2 1 2 2
2 1 7
```
output
```
0 2
```

input
```
5 7 2 5
1 5 3
1 4 8
5 3 2
4 5 6
2 5 6
3 4 4
2 3 3
```
output
```
3 3
```

C. Minimize the Danger

time limit per test: 2 seconds
memory limit per test: 1024 megabytes

You are in a city with N cities connected by M bi-directional roads. Each road has a danger level, where a higher number means the road is more dangerous.

You start in city 1 and need to go to every other city. The goal is to find the minimum danger level you would face to reach each city from city 1. The danger of a path is defined as the highest danger level of any road on that path.

For each city, find the minimum danger level of the path from city 1. If a city is not reachable from city 1, print -1. The danger of reaching city 1 is always 0.

Input

The first line contains two integers $N, M(2 \le N \le 2\cdot10^5, 1 \le M \le 3\cdot10^5)$ — the number of cities, the total number of roads.

The next M lines will contain three integers $u_i, v_i, w_i(1 \le u_i, v_i \le N, 1 \le w_i \le 10^6)$ — node u_i is connected to node v_i with a weight w_i.

Output

Output n integers, where i-th integer represents the minimum danger level you'd have to face in order to go from city 1 to city i.

Examples

input
```
4 3
2 1 3
2 3 5
3 4 3
```
output
```
0 3 5 5
```

input
```
6 5
1 2 3
1 4 5
1 6 2
2 6 3
4 6 1
```
output
```
0 3 -1 2 -1 2
```

input
```
2 1
2 1 5
```
output
```
0 5
```

input
```
5 7
1 5 3
1 4 2
5 3 1
5 4 6
5 2 5
3 4 4
3 2 8
```
output
```
0 5 3 2 3
```

D. Beautiful Path

time limit per test: 2 seconds
memory limit per test: 1024 megabytes

You are given an **directed** graph with N nodes and M edges. The graph contains no self-loops or multiple edges. The edges have no weight. The nodes are numbered from 1 to N and have a weight. You need to find the cost of a path, if there is any, from node S to node D with the minimum cost. The cost of a path is the sum of the weights of the nodes in that path.

Input

The first line contains four integers $N, M, S, D(2 \le N \le 2\times10^5, 1 \le M \le 3\times10^5, 1 \le S, D \le N)$ — the number of vertices, total number of edges, source, and destination.

In the next line, there will be N integers $w_1, w_2, w_3 \ldots w_n$ $(1 \le w_i \le 10^6)$ separated by spaces – representing the weights of each node.

The next M lines will contain two integers $u_i, v_i(1 \le u_i, v_i \le N)$ — there is an edge from the node u_i to the node v_i.

Output

If a valid path exists from S to D, print the minimum cost of the path. Otherwise, print -1.

Examples

input
```
4 3 1 2
3 4 5 4
1 2
3 2
4 3
```
output
```
7
```

input
```
6 5 5 3
3 3 4 3 4 1
1 2
4 1
1 6
6 2
4 6
```
output
```
-1
```

input
```
2 1 1 1
7 6
2 1
```
output
```
7
```

input
```
5 7 3 5
3 8 2 6 6
1 5
1 4
5 3
4 5
2 5
3 4
2 3
```
output
```
14
```

E. Parity Edges

time limit per test: 1.5 seconds
memory limit per test: 1024 megabytes

You are given a **directed weighted** graph with N nodes and M edges. The nodes are numbered from 1 to N. The graph contains no self-loops or multiple edges.

Your task is to find the shortest distance from node 1 to node N, with an additional constraint: the path cannot contain two consecutive edges with the same parity (i.e., both even or both odd). If no such path exists, print -1.

Input

The first line contains two integers $N, M(2 \le N \le 2\times10^5, 1 \le M \le 3\times10^5)$ — the number of vertices, total number of edges.

The second line contains M integers $u_1, u_2, u_3 \ldots u_m$ $(1 \le u_i \le N)$ — where the i-th integer represents the node that is one endpoint of the i-th edge.

The third line contains M integers $v_1, v_2, v_3 \ldots v_m$ $(1 \le v_i \le N)$ — where the i-th integer represents the node that is the other endpoint of the i-th edge.

The fourth line contains M integers $w_1, w_2, w_3 \ldots w_m$ $(1 \le w_i \le 10^6)$ — where the i-th integer represents the weight of the i-th edge.

The i-th edge of this graph is from the i-th node in the second line to the i-th node in the third line.

Output

Print a single integer — the minimum distance from node 1 to node N following the given constraint. If there is no valid path, print -1.

Examples

input
```
4 3
1 3 2
4 4 3
3 4 5
```
output
```
3
```

input
```
6 5
1 4 1 6 4
2 1 6 2 6
3 3 4 3 4
```
output
```
4
```

input
```
2 1
2
1
7
```
output
```
-1
```

input
```
5 7
1 1 4 5 2 3 2
4 5 3 4 4 5 3
3 8 2 6 6 4 3
```
output
```
8
```

F. Shortest Path Revisited

time limit per test: 3 seconds
memory limit per test: 1024 megabytes

You are given a **bidirectional weighted** graph with N nodes and M edges. The nodes are numbered from 1 to N. The graph contains no self-loops or multiple edges.

There is a source and a destination. Your task is to find the cost of the second shortest path from the source node to the destination node. If no such path exists, print -1.

Note: Second shortest path will be strictly greater than the shortest path

Input

The first line contains four integers $N, M, S, D(2 \le N \le 2\times10^5, 1 \le M \le 3\times10^5, 1 \le S, D \le N)$ — the number of vertices, total number of edges, source, and destination.

The next M lines will contain three integers $u_i, v_i, w_i(1 \le u_i, v_i \le N, 1 \le w_i \le 10^6)$ — there is an edge between the node u_i and the node v_i with a weight w_i.

Output

- If a valid path exists from S to D, print the cost of the second shortest path from S to D.
- If no such path exists, print -1.

Examples

input
```
4 3 2 3
2 1 3
2 3 5
3 4 3
```
output
```
11
```

input
```
6 5 3 4
1 2 3
1 4 5
1 6 2
2 6 3
4 6 1
```
output
```
-1
```

input
```
2 1 2 2
2 1 5
```
output
```
10
```

input
```
5 7 2 5
1 5 3
1 4 2
5 3 1
5 4 6
5 2 5
3 4 4
3 2 8
```
output
```
7
```