

Minimum Penalty Path

Consider an undirected graph containing N nodes and M edges. Each edge M_i has an integer *cost*, C_i , associated with it.

The *penalty* of a path is the *bitwise OR* of every edge cost in the path between a pair of nodes, A and B . In other words, if a path contains edges M_1, M_2, \dots, M_k , then the penalty for this path is C_1 **OR** C_2 **OR** ... **OR** C_k .

Given a graph and two nodes, A and B , find the path between A and B having the *minimal possible penalty* and print its penalty; if no such path exists, print -1 to indicate that there is no path from A to B .

Note: Loops and multiple edges are allowed. The bitwise OR operation is known as **or** in Pascal and as $|$ in C++ and Java.

Input Format

The first line contains two space-separated integers, N (the number of nodes) and M (the number of edges), respectively.

Each line i of the M subsequent lines contains three space-separated integers U_i , V_i , and C_i , respectively, describing edge M_i connecting the nodes U_i and V_i and its associated penalty (C_i).

The last line contains two space-separated integers, A (the starting node) and B (the ending node), respectively.

Constraints

- $1 \leq N \leq 10^3$
- $1 \leq M \leq 10^4$
- $1 \leq C_i < 1024$
- $1 \leq U_i, V_i \leq N$
- $1 \leq A, B \leq N$
- $A \neq B$

Output Format

Print the minimal penalty for the optimal path from node A to node B ; if no path exists from node A to node B , print -1 .

Sample Input

```
3 4
1 2 1
1 2 1000
2 3 3
1 3 100
1 3
```

Sample Output

Explanation

The optimal path is $1 \rightarrow 2 \rightarrow 3$.

$C_{(1,2)} = 1$ and $C_{(2,3)} = 3$.

The penalty for this path is: $1 \text{ OR } 3 = 3$, so we print 3 .