Supplies

Rar the Cat wants to get some supplies from some faraway land and return back home. Since the 'faraway land' is so far away, there might or might not be direct transportation services/routes to that location from Rar's house.

Rar the Cat has investigated the bus routes from his house to the faraway land and has counted a total of **N** bus stops/interchanges labelled from **0** to **N-1** and **E** bus routes. Each bus route costs a certain amount of money (**C**) and ferries people from bus stop **A** to bus stop **B**.

A person (or cat) can get from one bus stop to another via buses. However, since **some routes aren't bi-directional**, the cost of travelling from bus stop 1 to bus stop 2 **could be different** from travelling from bus stop 2 to bus stop 1.

Rar the Cat wants to know how much money would he need to spend on transportation if he decides to get the supplies **and return back home**. If it costs too much, he can order it from an online retailer.

No 2 routes will have the same **A** and **B**. In other words, there will not be two buses that ferries people from the same starting bus stop to the same destination bus stop.

Input
The first line of input consists of 4 integers: **N**, **E**, **H** and **T**. **H** represents which bus stop Rar the Cat will be starting in and **T** represents which bus stop Rar the Cat wants to obtain supplies from.

Subsequently, **E** lines follow with 3 integers each: **A**, **B** and **C**. Each line describes a bus route, meaning there is a bus service that services from bus stop **A** to bus stop **B** and costs **$C**.

Output
Output a single integer, the amount of money Rar the Cat would spend on transportation if he decides to go get the supplies and returns home.

If it is impossible (no route) to get the supplies and return home, output -1 instead.

Limits

- 0 < **N** ≤ 1000, 0 ≤ **E** ≤ **N*(N-1)**
- 0 ≤ **A**, **B**, **H**, **T** < **N**
- 0 ≤ **C** ≤ 10000 for all bus routes.

Sample Testcase 1

Sample Input (**supplies1.in**)	Sample Output (**supplies1.out**)

4 8 0 3
0 1 1
1 0 4
1 3 5
1 2 3
2 0 3
2 3 1
2 1 8
3 2 6
	14

Explanation for Sample Testcase 1

It takes $5 to get from bus stop 0 to bus stop 3 using the route 0 -> 1, 1 -> 2 and 2 -> 3. It takes $9 to get from bus stop 3 to bus stop 0 using the route 3 -> 2 and 2 -> 0. The total cost is hence $14.

Sample Testcase 2

Sample Input (**supplies2.in**)	Sample Output (**supplies2.out**)

4 8 1 3
0 1 1
1 0 4
1 3 5
1 2 3
2 0 3
2 3 1
2 1 8
3 2 6
	14

Explanation for Sample Testcase 2

It takes $4 to get from bus stop 1 to bus stop 3 using the route 1 -> 2 and 2 -> 3. It takes $10 to get from bus stop 3 to bus stop 1 using the route 3 -> 2, 2 -> 0, 0 -> 1. The total cost is hence $14.

Sample Testcase 3

Sample Input (**supplies3.in**)	Sample Output (**supplies3.out**)

5 5 1 3
0 1 5
1 2 3
2 3 2
3 4 1
4 2 4
	-1

Explanation for Sample Testcase 2

Although it is possible to get from bus stop 1 to bus stop 3 using $5 and the route 1 -> 2 and 2 -> 3, there is no way to get from bus stop 3 to bus stop 1.

Notes:

1. You should develop your program in the subdirectory **ex3** and use the skeleton java file provided. You should not create a new file or rename the file provided.
2. You are free to define your own helper methods and classes (or remove existing ones).
3. Please be reminded that the marking scheme is:
    a. Public Test Cases (1%) - 1% for passing **all** test cases, 0% otherwise
    b. Hidden Test Cases (1%) - Partial scoring depending on test cases passed
    c. Manual Grading (1%)
        i. Overall Correctness (correctness of algorithm, severity of bugs)
        ii. Coding Style (meaningful comments, modularity, proper indentation, meaningful method and variable names)
4. Your program will be tested with a time limit of not less than **2 sec** on Codecrunch.

Skeleton File – Supplies.java

You are given the skeleton file `Supplies.java`. You should see a non-empty file when you open the skeleton file. Otherwise, you might be in the wrong working directory.

You should see the following contents when you open the skeleton file:

```
import java.util.*;
public class Supplies {
    private void run() {
        //implement your "main" method here
    }
    public static void main(String[] args) {
        Supplies newSupplies = new Supplies();
        newSupplies.run();
    }
}
```