Problem C - Circlular Shopping

Daniel realized that there were no more disposable plates for the next ACM practice (where they would be ordering pizza as usual), so his friend Jason offered him a ride to the mall.

The mall is a pretty complicated place, but it can essentially be modelled by an undirected graph where the vertices are stores that can potentially sell disposable plates, and edges are stores that are walkable from one another. Jason can drop Daniel off at any store, where he'll wait in the car for Daniel to return.

Daniel is pretty frugal so he wants to walk around the mall to find the cheapest plates. Daniel wants to check out at least three stores, but he doesn't mind looking into more stores. To be efficient, he also doesn't want to pass by the same store more than once, except for the store where Jason dropped Daniel off. Formally a valid route of this form is a *cycle*.

Daniel absolutely hates walking long distances between consecutive stores because he can't stand the crowd. Let's call the longest contiguous period of time Daniel needs to be walking in the crowd between stores in a cycle C the *crowd time* of the cycle C. Help Daniel find the minimum crowd time among all cycles he can take.

Input

The first line contains an integer T denoting the number of test cases.

For each test case, the first line contains two space seperated integers $1 \leq n \leq 10^5$ denoting the number of stores and $1 \leq m \leq 10^6$ denoting the number of pairs of stores that are walkable from one to the other.

The next m lines contain three space seperated integers per line $1 \leq a, b \leq n$ denoting the index of the store and $1 \leq t \leq 10^9$ denoting the time it takes to walk between store a and b. It is guaranteed that there is at most one edge between any such a and b.

Output

For each test case ouput one line containing the minimum crowd time in any cycle Daniel can take, or `-1` if such a route doesn't exist.

Sample Input

```
3
4 4
1 2 3
2 3 5
3 1 2
1 4 12
4 5
1 2 1
1 3 1
2 4 1
3 4 1
1 4 1
2 1
1 2 1
```

Sample Output

```
5
1
-1
```