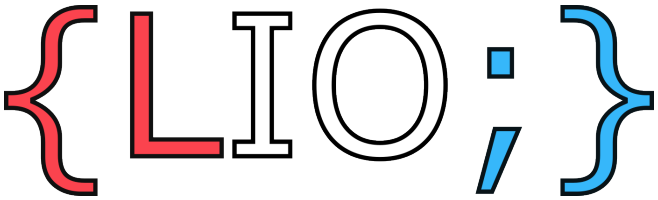
Lëtzebuerger Informatiksolympiad 2024

Finals

Task descriptions

Instructions

- The allowed programming language is C/C++.
- All the programs must be realized in the form of a console application. For instructions how to realize a console application in the allowed programming languages, please refer to the remarks on the site `www.infosolympiad.lu` under the heading *The tasks*.
- Under the input of the program is meant either the direct entry of data from the keyboard or the redirection from a text file in console mode. Under output of the program is meant either the direct display of data to the screen or the redirection to a text file in console mode.
- The formats of the input and output data shown in the execution examples must absolutely be respected.
- For testing, submitting and evaluating a program, the source file with a file extension `c/cpp` must be uploaded to the automated online judge CMS (Contest Management System), accessible via the homepage `www.infosolympiad.lu` or directly via the URL `http://158.64.46.20:81`. Please use your personal login (username and password) to access your account on the CMS. The filename of the single source file should be the same than the task name. Please refer to the CMS for technical details on how to test and submit a program.
- Please refer to the CMS for technical details like time limits and memory limits as well as compilation commands.
- You have the right to ask questions via the CMS, but the answers will not teach you how to use a programming language nor tell you how to solve the tasks by using a specific algorithm. The questions should be in relation with the CMS or should treat clarification issues concerning the task descriptions.



Catching Pokémon

Description

Ketti has recently started playing Pokémon and wants to catch all P Pokémon. However, she has lost her charging cable so her battery will run out in 10^9 minutes. Help her find the shortest time in which she can catch all Pokémon, or determine if it's impossible to catch all Pokémon before the battery runs out.

There are N islands (numbered from 0 to $N-1$) on the map and M bridges connecting them. Each bridge connects two islands u and v and it takes w minutes to cross this bridge. Bridges can be travelled in both directions. Two islands will be connected by at most one bridge. At the beginning of the game, Ketti is on island 0.

The P Pokémon are numbered from 0 to $P-1$. Ketti knows exactly where and when each Pokémon will show up on the map [1]. She has Q predictions, each of the following form:

Pokémon p will be catchable on island k at time t.

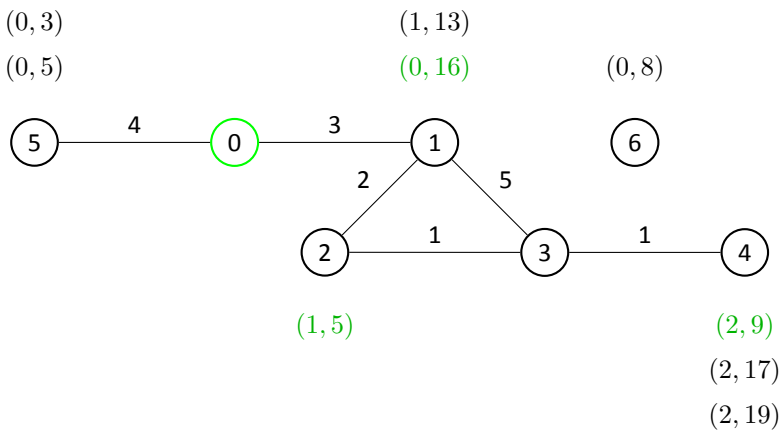
If Ketti is on an island at the same time as a Pokémon, she can catch it. Catching a Pokémon takes her 1 minute.

Based on these predictions, calculate the minimum time needed to catch all Pokémon, or determine if it's impossible.

Task

Find the shortest time in which she can catch all Pokémon, or determine if it's impossible.

Example



The figure above illustrates the testcase represented in the Execution example. Near each island you can find a (p, t) tuple representing that a Pokemon p appears at time t.

The optimal solution starts by going to island 2 (in 5 minutes) and catching the Pokemon 1 at time $t = 5$, and can leave at $t = 6$. Then Ketti can go to island 4 (in 2 minutes) where she catches the Pokemon 2 and can leave at $t = 10$. Finally she can go to island 1 (through the $4, 3, 1$ path), catch Pokemon 0 at $t = 16$ and finish catching every Pokemon by $t = 17$.

Constraints

- $1 \leq N \leq 10^4$
- $1 \leq M \leq 2 \cdot 10^4$
- $1 \leq P \leq 8$
- $1 \leq Q \leq 10^5$

[1] She has a PhD in Pokémon Behaviour.

Input and output of program

Input data

The first line contains three integers N (the number of islands) and M (the number of bridges), and P (the number of Pokémon).

The next M lines contain the description of bridges, each containing three integers:

u_i, v_i and w_i ($u_i \neq v_i, 0 \leq u_i, v_i < N, 1 \leq w_i \leq 10^5$ - denoting there is a bridge between u_i and v_i which takes w_i seconds to cross.

The next line contains one integer Q (the number of predictions).

The next Q lines contain the description of each prediction, each containing three integers p_j, k_j and t_j - denoting Pokémon p_j will be on island k_j at time t_j. ($0 \leq p_j < P,\ 0 \leq k_j < n,\ 0 \leq t_j \leq 10^9$)

Output data

Output the minimum time to catch all Pokémon if it is possible based on the predictions. If it is not possible output -1

Execution example

Input

```
7 6 3
0 1 3
0 5 4
1 2 2
1 3 5
2 3 1
3 4 1
9
0 5 3
1 2 5
0 5 5
2 4 9
1 1 13
0 6 8
2 4 17
0 1 16
2 4 19
```

Output

```
17
```

Distribution of points

Subtask	Points	Constraints/Description
1	5	$N = 2$
2	10	$P = 1$
3	10	$M = N - 1$, Bridge i connects island i and i+1 for $0 \leq i < N - 1$ with $w_i = 1$
4	10	$N \leq 100$
5	15	No additional constraints

Technical constraints

Task name	pokemon
Input file	standard input
Output file	standard input
Time limit	3 seconds
Memory limit	256 megabytes

Euclidean Sort

Description

You are given a line with N marked points, which we denote by $1, 2, \ldots, N$. Note however that the points with this numbering are not necessarily ordered. In other words it is not necessarily the case that for any three points $p, q, r \in \{1, \ldots, N\}$ with $p < q < r$ that q lies in-between p and r. It is your task to sort the points on the line.

Task

This is an interactive task. You can probe the arrangement of the points using the function cmp(p, q, r). It will return the point that is in the middle of the two other points. You can then change the arrangement of points by invoking swap(p, q), which will exchange the points p and q. Your score depends on the number of times you invoke the function cmp, which we call the number of queries Q. You have accomplished your task if after number of swaps, the points are arranged in such a way that for any three points $p, q, r \in \{1, \ldots, N\}$ with $p < q < r$, the point q lies in-between p and r.

Example

Suppose that $N = 4$. You make the following function calls:

- cmp$(1, 2, 3)$, which returns 2.
- cmp$(1, 2, 4)$, which returns 4
- cmp$(1, 3, 4)$, which returns 4
- cmp$(2, 1, 3)$, which returns 2
- cmp$(2, 1, 4)$, which returns 2
- cmp$(2, 3, 4)$, which returns 4

You can then call the command swap$(3, 4)$ to order the points correctly. Now the points are sorted. You can check for yourself that the initial arrangement must have been 4312 or 1243. In particular, calling the following series of swap commands also leads to a correct sorting.

- swap$(1, 3)$
- swap$(3, 4)$
- swap$(2, 3)$

Constraints

- $3 \leq N \leq 500$
- $0 \leq Q \leq 10.000$

Input and output of program

This task is interactive. You need to implement the function sort in the skeleton implementation, which you can find on CMS.

Use of sample grader

You can find a sample grader in the task description on CMS as well as the required compilation commands.

Input data

The first line contains two integers N and Q, the number of points to be sorted and the maximum allowed number of queries. The second line contains the N integers in some order, describing the initial arrangement of the points on the line.

Output data

The output will consist of a single line. The first word of the output will be either ”sorted” or ”unsorted”, depending on whether your program successfully sorted the points. The second word will be either ”accepted” in case your program stayed within the limit of Q queries, or ”rejected” if it exceeded the number of allowed queries. The line will be concluded by an integer number which is number of times your program queried cmp. Here is the sample input corresponding to the example above, and some possible outputs by the sample grader.

Input

```
4 18
4 3 1 2
```

Possible Output

```
sorted accepted 15
```

Possible Output

```
sorted rejected 20
```

Possible Output

```
unsorted accepted 0
```

Distribution of points

Subtask	Points	Constraints/Description
1	3	$N \leq 4$
2	8	$N \leq 100$
3	11	$Q \leq 10000$
4	18	$Q \leq 5000$
5	10	$Q \leq 3000$

Technical constraints

Task name	euclideansort
Input file	none
Output file	none
Time limit	1 second
Memory limit	256 megabytes