max planck institut
informatik

Competitive Programming

Saarland University — Summer Semester 2020

Julian Baldus, Markus Bläser, Karl Bringmann, Marian Dietz, Simon Schwarz,
Christoph Weidenbach, Dominic Zimmer

Midterm — Problem Set

June 20, 2020

Please submit solutions to the problems in our judge system, available from within the VM at `https://compro.mpi-inf.mpg.de/`.
You can find your credentials on your personal status page in our CMS.

Problem	airline	booster	closingtime			equation		lisgraph	
Points	10	15	10	10	10	15	10	10	10
Difficulty									
Time Limit	1s	2s	1s	1s	1s	1s	1s	2s	2s
Memory Limit	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB	2 GB

Please note:

- Your solution will be judged immediately after submitting. This may take some time, depending on the current server load.
- You can submit as many times as you want. However, don't abuse the server or try to extract the secret test cases.
- If your solution is **accepted**, you will receive the points specified in the table above.
- If you get **another verdict**, we will look at your latest submission for the corresponding subtask. You will receive partial points if we can understand your code and it contains minor errors only. Otherwise, you will receive 0 points.

Airline (10 points)

Problem ID: airline
Time limit: 1 second

You are operating an airline. Today, there are n possible flights that you can offer. Each flight i has a profit p_i assigned to it, meaning that if you realize flight i, you will make a profit of p_i euros. Some flights are not booked out and, therefore, have a negative profit.

Today, k flight slots have been assigned to you. This means that you can realize only k flights today. Since you loose your slots permanently if you do not use them, you must realize exactly k flights today - no more, and no less. Note that you can offer each flight in your list only once.

You want to pick the k flights that you will realize today out of the list of n possible flights. Note that you may also have to pick flights with negative profit to occupy all your starting slots.

What is the maximal profit you can reach today?

Input

The first line contains two integers n ($1 \leq n \leq 200\,000$) and k ($1 \leq k \leq n$), the number of possible flights and the number of starting slots. The second line contains n integers p_i ($-10^9 \leq p_i \leq 10^9$).

Output

Print the maximal total profit you can make.

Sample Input 1

```
7 3
47 28 3 1 -10 38 1000
```

Sample Output 1

```
1085
```

Sample Input 2

```
7 5
36 -3 -3 -150 46 5 -17
```

Sample Output 2

```
81
```

Sample Input 3

```
3 3
1000000000 999999999 1000000000
```

Sample Output 3

```
2999999999
```

Space for your wonderful thoughts

Booster (15 points)

Problem ID: booster

There is a straight line with n positions. Position 1 is the leftmost position, position n is the rightmost position. You are standing at position 1 and want to get to position n in the shortest time possible. The are two possibilities how you can move:

- At any point, you can walk one step to the right (e.g. from i to $i + 1$). This takes exactly one second.
- There are k boosters, each of which starts at a position s_i and boosts you to position e_i with $1 \leq s_i < e_i \leq n$. If you are at position p, you can use (but don't have to) any booster i with $s_i = p$ to get boosted to e_i. The boosting process takes exactly one second.

Input

The first line contains two integers n ($2 \leq n \leq 10^6$) and k ($1 \leq k \leq 100\,000$). The following k lines each contain two integers s_i and e_i ($1 \leq s_i < e_i \leq n$).

Output

Print the minimum number of seconds needed to get from position 1 to position n.

Sample Input 1

```
20 5
2 8
2 6
5 9
11 16
18 20
```

Sample Output 1

```
9
```

Sample Input 2

```
10 2
2 4
3 9
```

Sample Output 2

```
4
```

Sample Input 3

```
1000000 3
4 1000000
1 5
1 3
```

Sample Output 3

```
3
```

Space for your wonderful thoughts

Closing Time (30 points)

Problem ID: closingtime

The *Ladenschlussgesetz* [law that governs the hours of trading] of the Saarland is very strict. No supermarket is allowed to be open after 20 o'clock. You just discovered that you are running out of food and urgently need something to eat from a supermarket. Unfortunately, it is already almost 20:00. Therefore, you are willing to ask your friends if they would go shopping for you. Probably one of your friends is living very close to a supermarket and can reach it in time.

We can model the city as a *weighted, directed* graph. Saarbrücken consists of n locations and m one-way streets between them. Street i leads from location s_i to location t_i. Passing through street i takes v_i minutes.

You have k_1 friends and there are k_2 supermarkets in Saarbrücken. You know the locations of all your friends and the locations of all supermarkets. What is the minimum time needed for any friend to reach the closest supermarket?

Subtasks

- **Subtask 1**  (10 points) $k_1 = k_2 = 1$, i.e. there is only one friend and one supermarket.
- **Subtask 2**  (10 points) $k_1 = 1$ and $1 \leq k_2 \leq n$, i.e. there is only one friend, but many supermarkets.
- **Subtask 3**  (10 points) $1 \leq k_1, k_2 \leq n$, i.e. there may be many friends and supermarkets.

Input

The first line contains four integers: n, m, k_1 and k_2 ($1 \leq n \leq 100\,000$, $1 \leq m \leq 200\,000$, $1 \leq k_1, k_2 \leq n$).

The second line contains k_1 integers $f_1, \ldots, f_{k_1}$ ($1 \leq f_i \leq n$ for $1 \leq i \leq k_1$), the positions of your friends.

The third line contains k_2 integers $g_1, \ldots, g_{k_2}$ ($1 \leq g_j \leq n$ for $1 \leq j \leq k_2$), the positions of the supermarkets.

The following m lines contain three integers s_i, t_i, v_i each ($1 \leq s_i, t_i \leq n$, $s_i \neq t_i$, $1 \leq v_i \leq 10^9$). They represent a street from node s_i to t_i, which takes v_i minutes to pass. It is guaranteed that for any two locations u and v, there is at most one street from u to v. Additionally, all supermarkets are at distinct locations and all friends are at distinct locations, although there may be a friend and a supermarket at the same location.

Output

Print the shortest time needed for any friend to reach any supermarket. If there is no friend who can reach a supermarket, print `impossible` instead.

Sample Inputs

The first three samples can occur in all subtasks. Sample 4 and 5 can only occur in subtask 2 and 3. The last sample can only occur in subtask 3.

Sample Input 1	Sample Output 1
4 5 1 1 4 2 1 3 1 4 1 2 4 3 4 3 1 1 3 2 5	8

Sample Input 2

```
6 5 1 1
1
6
1 2 1000000000
2 3 1000000000
3 4 1000000000
4 5 1000000000
5 6 1000000000
```

Sample Output 2

```
5000000000
```

Sample Input 3

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

Sample Output 3

```
impossible
```

Sample Input 4

```
6 6 1 2
6
4 1
6 2 1
2 3 3
3 4 2
4 2 1
6 5 3
5 1 5
```

Sample Output 4

```
6
```

Sample Input 5

```
3 3 1 2
2
2 1
1 2 10
2 3 10
3 1 10
```

Sample Output 5

```
0
```

Sample Input 6

```
10 6 3 2
1 5 10
3 8
1 2 100
2 3 1
5 3 200
10 2 5
3 8 1
4 8 2
```

Sample Output 6

```
6
```

Space for your wonderful thoughts

Equations

Problem ID: equation
Time limit: 1 second

You are given an expression $f(\texttt{X})$ that has been generated by the rule $\langle\text{tree}\rangle$ of the following grammar:

$$\langle\text{tree}\rangle \;:=\; \text{“(”}\ \langle\text{tree}\rangle\ \text{“+”}\ \langle\text{tree}\rangle\ \text{“)”} \;\mid\; \text{“(”}\ \langle\text{tree}\rangle\ \text{“}\star\text{”}\ \langle\text{tree}\rangle\ \text{“)”} \;\mid\; \langle\text{nat}\rangle \;\mid\; \text{“X”}$$
$$\langle\text{nat}\rangle \;:=\; \text{a natural number between 1 and 9}$$

Note that the expression cannot contain constants with more than one digit.

The expression $f(a)$ can be evaluated by replacing all occurences of X with a and then calculating the result of the resulting term. Given two integers a_{max} and y, your task is to find an integer a with $1 \leq a \leq a_{max}$ and $f(a) = y$.

Subtasks

- **Subtask 1**  (15 points) $1 \leq a_{max} \leq 10$
- **Subtask 2**  (10 points) $1 \leq a_{max} \leq 10^{18}$

Input

The first line of the input contains two integers a_{max} and y ($1 \leq a_{max}, y \leq 10^{18}$). The second line contains the expression $f(\texttt{X})$ as a string of at most 10^5 characters. It contains at least one X and you may assume $f(a_{max}) \leq 10^{18}$.

Output

Print an integer a with $1 \leq a \leq a_{max}$ and $f(a) = y$ or print `impossible` if there is no such a.

Warning

If you attempt to write a solution in Python which calls `eval` or `exec` on the string representing f, your program will exceed the memory limit and receive a `RUNTIME ERROR` verdict. This is due to a long-standing bug in the Python parser.

Sample Inputs

The first three samples may occur in both subtasks. The last sample can occur in the second subtask only.

Sample Input 1

```
10 42
+(*(X,009),*(X,+(X,002)))
```

Sample Output 1

```
3
```

Sample Input 2

```
10 6
+(001,+(*(002,X),004))
```

Sample Output 2

```
impossible
```

Sample Input 3

```
10 987
+(*(X,123),987)
```

Sample Output 3

```
impossible
```

Sample Input 4	Sample Output 4
1000000000000000000 734258734658937645 X	734258734658937645

Space for your wonderful thoughts

LIS on a Graph (20 points)

Problem ID: lisgraph

You are given a directed graph with n nodes and m edges. The nodes are numbered from 1 to n. Every node i has a value, denoted by v_i.

A path is a finite sequence of nodes $u_1, \ldots, u_k$ such that there is an edge from u_i to u_{i+1} for all $1 \leq i \leq k-1$. k can be arbitrarily large and nodes can be visited more than once. The value sequence of a path $p = u_1, \ldots, u_k$ is $v(p) = v_{u_1}, \ldots, v_{u_k}$. In other words, a value sequence consists of the values of the nodes forming a path, in the same order as the corresponding nodes.

Consider all possible value sequences in our graph. We are interested in a value sequence having the longest possible Longest strictly Increasing Subsequence (LIS). You task is to calculate the length of its LIS (i.e. find the length of the longest LIS over all value sequences).

As a reminder: An increasing subsequence of a sequence $a_1, \ldots, a_l$ is a sequence $a_{i_1}, \ldots, a_{i_r}$ with $i_1 < \cdots < i_r$ and $a_{i_1} < \cdots < a_{i_r}$. A longest increasing subsequence is an increasing subsequence of maximum length.

Subtasks

- **Subtask 1**  (10 points) The given graph is guaranteed to be a directed acyclic graph.
- **Subtask 2**  (10 points) No further constraints.

Input

The first line contains two integers n ($1 \leq n \leq 5000$) and m ($1 \leq m \leq 10\,000$).

The next line contains n integers v_i ($1 \leq v_i \leq 10^9$).

The following m lines each contain two integers s_i and d_i ($1 \leq s_i, d_i \leq n$), representing an edge from s_i to d_i.

It is guaranteed that the graph contains no selfloops or multiedges.

Output

Print the length of the longest possible LIS.

Sample Inputs

The first sample may appear in both subtasks, while the second sample can occur in subtask 2 only.

Sample Input 1	Sample Output 1
7 7 3 2 2 6 4 5 7 3 2 3 1 2 4 1 4 4 5 5 6 4 7	4

Sample Input 2	Sample Output 2
9 10 4 3 2 1 3 4 5 6 2 1 2 2 3 3 4 4 1 1 5 5 6 5 7 6 8 7 8 8 9	6

Space for your wonderful thoughts