Codeforces Round #332 (Div. 2)

A. Patrick and Shopping

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Today Patrick waits for a visit from his friend Spongebob. To prepare for the visit, Patrick needs to buy some goodies in two stores located near his house. There is a d_1 meter long road between his house and the first shop and a d_2 meter long road between his house and the second shop. Also, there is a road of length d_3 directly connecting these two shops to each other. Help Patrick calculate the minimum distance that he needs to walk in order to go to both shops and return to his house.

Patrick always starts at his house. He should visit both shops moving only along the three existing roads and return back to his house. He doesn't mind visiting the same shop or passing the same road multiple times. The only goal is to minimize the total distance traveled.

Input

The first line of the input contains three integers d_1, d_2, d_3 ($1 \le d_1, d_2, d_3 \le 10^8$) — the lengths of the paths.

- d_1 is the length of the path connecting Patrick's house and the first shop;
- d_2 is the length of the path connecting Patrick's house and the second shop;
- d_3 is the length of the path connecting both shops.

Output

Print the minimum distance that Patrick will have to walk in order to visit both shops and return to his house.

Examples

input
10 20 30
output
60

input
1 1 5
output
4

Note

The first sample is shown on the picture in the problem statement. One of the optimal routes is: house ⟶ first shop ⟶ second shop ⟶ house.

In the second sample one of the optimal routes is: house ⟶ first shop ⟶ house ⟶ second shop ⟶ house.

B. Spongebob and Joke

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

While Patrick was gone shopping, Spongebob decided to play a little trick on his friend. The naughty Sponge browsed through Patrick's personal stuff and found a sequence $a_1, a_2, \ldots, a_m$ of length m, consisting of integers from 1 to n, not necessarily distinct. Then he picked some sequence $f_1, f_2, \ldots, f_n$ of length n and for each number a_i got number $b_i = f_{a_i}$. To finish the prank he erased the initial sequence a_i.

It's hard to express how sad Patrick was when he returned home from shopping! We will just say that Spongebob immediately got really sorry about what he has done and he is now trying to restore the original sequence. Help him do this or determine that this is impossible.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 100\,000$) — the lengths of sequences f_i and b_i respectively.

The second line contains n integers, determining sequence $f_1, f_2, \ldots, f_n$ ($1 \le f_i \le n$).

The last line contains m integers, determining sequence $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le n$).

Output

Print "Possible" if there is exactly one sequence a_i, such that $b_i = f_{a_i}$ for all i from 1 to m. Then print m integers $a_1, a_2, \ldots, a_m$.

If there are multiple suitable sequences a_i, print "Ambiguity".

If Spongebob has made a mistake in his calculations and no suitable sequence a_i exists, print "Impossible".

Examples

input
3 3 3 2 1 1 2 3
output
Possible 3 2 1

input
3 3 1 1 1 1 1 1
output
Ambiguity

input
3 3 1 2 1 3 3 3
output
Impossible

Note

In the first sample 3 is replaced by 1 and vice versa, while 2 never changes. The answer exists and is unique.

In the second sample all numbers are replaced by 1, so it is impossible to unambiguously restore the original sequence.

In the third sample $f_i \neq 3$ for all i, so no sequence a_i transforms into such b_i and we can say for sure that Spongebob has made a mistake.

C. Day at the Beach

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

One day Squidward, Spongebob and Patrick decided to go to the beach. Unfortunately, the weather was bad, so the friends were unable to ride waves. However, they decided to spent their time building sand castles.

At the end of the day there were n castles built by friends. Castles are numbered from 1 to n, and the height of the i-th castle is equal to h_i. When friends were about to leave, Squidward noticed, that castles are not ordered by their height, and this looks ugly. Now friends are going to reorder the castles in a way to obtain that condition $h_i \le h_{i+1}$ holds for all i from 1 to $n - 1$.

Squidward suggested the following process of sorting castles:

- Castles are split into *blocks* — groups of **consecutive** castles. Therefore the block from i to j will include castles $i, i + 1, \ldots, j$. A block may consist of a single castle.
- The partitioning is chosen in such a way that every castle is a part of **exactly** one block.
- Each block is sorted independently from other blocks, that is the sequence $h_i, h_{i+1}, \ldots, h_j$ becomes sorted.
- The partitioning should satisfy the condition that after each block is sorted, the sequence h_i becomes sorted too. This may always be achieved by saying that the whole sequence is a single block.

Even Patrick understands that increasing the number of blocks in partitioning will ease the sorting process. Now friends ask you to count the maximum possible number of blocks in a partitioning that satisfies all the above requirements.

Input

The first line of the input contains a single integer n ($1 \le n \le 100\,000$) — the number of castles Spongebob, Patrick and Squidward made from sand during the day.

The next line contains n integers h_i ($1 \le h_i \le 10^9$). The i-th of these integers corresponds to the height of the i-th castle.

Output

Print the maximum possible number of blocks in a valid partitioning.

Examples

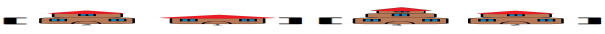
input
3 1 2 3
output
3

input
4 2 1 3 2
output
2

Note

In the first sample the partitioning looks like that: [1][2][3].

In the second sample the partitioning is: [2, 1][3, 2]



D. Spongebob and Squares

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Spongebob is already tired trying to reason his weird actions and calculations, so he simply asked you to find all pairs of n and m, such that there are exactly x distinct squares in the table consisting of n rows and m columns. For example, in a 3×5 table there are 15 squares with side one, 8 squares with side two and 3 squares with side three. The total number of distinct squares in a 3×5 table is $15 + 8 + 3 = 26$.

Input

The first line of the input contains a single integer x ($1 \le x \le 10^{18}$) — the number of squares inside the tables Spongebob is interested in.

Output

First print a single integer k — the number of tables with exactly x distinct squares inside.

Then print k pairs of integers describing the tables. Print the pairs in the order of increasing n, and in case of equality — in the order of increasing m.

Examples

input
26

output
6 1 26 2 9 3 5 5 3 9 2 26 1

input
2

output
2 1 2 2 1

input
8

output
4 1 8 2 3 3 2 8 1

Note

In a 1×2 table there are 2 1×1 squares. So, 2 distinct squares in total.

In a 2×3 table there are 6 1×1 squares and 2 2×2 squares. That is equal to 8 squares in total.

E. Sandy and Nuts

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Rooted tree is a connected graph without any simple cycles with one vertex selected as a root. In this problem the vertex number 1 will always serve as a root.

Lowest common ancestor of two vertices u and v is the farthest from the root vertex that lies on both the path from u to the root and on path from v to the root. We will denote it as $LCA(u, v)$.

Sandy had a rooted tree consisting of n vertices that she used to store her nuts. Unfortunately, the underwater storm broke her tree and she doesn't remember all it's edges. She only managed to restore m edges of the initial tree and q triples a_i, b_i and c_i, for which she supposes $LCA(a_i, b_i) = c_i$.

Help Sandy count the number of trees of size n with vertex 1 as a root, that match all the information she remembered. If she made a mess and there are no such trees then print 0. Two rooted trees are considered to be distinct if there exists an edge that occur in one of them and doesn't occur in the other one.

Input

The first line of the input contains three integers n, m and q ($1 \le n \le 13, 0 \le m < n, 0 \le q \le 100$) — the number of vertices, the number of edges and LCA triples remembered by Sandy respectively.

Each of the next m lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n, u_i \neq v_i$) — the numbers of vertices connected by the i-th edge. It's guaranteed that this set of edges is a subset of edges of some tree.

The last q lines contain the triplets of numbers a_i, b_i, c_i ($1 \le a_i, b_i, c_i \le n$). Each of these triples define $LCA(a_i, b_i) = c_i$. It's **not guaranteed** that there exists a tree that satisfy all the given LCA conditions.

Output

Print a single integer — the number of trees of size n that satisfy all the conditions.

Examples

input
4 0 0
output
16

input
4 0 1 3 4 2
output
1

input
3 1 0 1 2
output
2

input
3 0 2 2 3 2 2 3 1
output
0

input
4 1 2 1 2 2 2 2 3 4 2
output
1

Note

In the second sample correct answer looks like this:

In the third sample there are two possible trees:

In the fourth sample the answer is 0 because the information about LCA is inconsistent.
