01 Hr : 22 min : 22 sec

0 / 3 Attempted

Issues ?

Flag Question

Infinity Stones AvPpg

Problem Description

The mighty Thanos is obsessed with destroying the universe. But in order to do that he has to collect **k** infinity stones.
Thanos is tired of fighting. So he decides to behave civilised this time. As you are his most trusted ally, he gives you the task to bring him infinity stones.
In order to make an infinity stone we need to buy the corresponding nanocrystals. Nanocrystals can be combined to make infinity stone of some size. In simple words if you want to make the i^{th} infinity stone of size **Z** and you have the size of i^{th} nanocrystal **p** then amount of nano crystals required = $\lceil Z/p \rceil$.
Nanocrystal for one particular infinity stone can not be used to make another infinity stone. Each of the given k infinity stones shoudl be of the same size and as big as possible.
There are **N** shops $A_1\ A_2 \ldots A_n$ where you can by these infinity stones. Each shop A_i sells the j^{th} nanocrystal of size $A_{i,j}$. You can only take integral amount of stones from any shop.
But since the avengers are coming you will only have the time to visit any two shops. You can buy any type of nanocrytal in any amount from these **2** shops.
Each crystal of any type from any shop costs 1 dollar. Since fighting the war is financially challenging, Thanos only gave you **S** dollars.
Find the maximum size of all the inifinity stones you can make. Since the answer can be large, output modulo **1e9 + 7**

Problem Constraints

$1 <= N <= 10^5$

$1 <= k <= 7$

$1 <= A[i][j] <= 10^5$

ENG IN

$1 <= A[i][j] <= 10^5$

$1 <= S <= 10^9$

Input Format

First argument contains a 2D array A of size N * k.

Second argument contains an integer S.

Output Format

Example Input

Input1:

```
A =  [ [ 9,4,1],
       [2,5,10],
       [6,8,3]
     ]
S=10
```

Input2:

```
A =  [ [ 4,1],
       [1,4],
     ]
S=10
```

Example Output

Output1:

24

Output1:

20

Example Explanation

Explaination1:

We can use shop 2 and 3... total cost for size = 24. 24/6 + 24/8 + ceil(24/10) => 4 + 3 + 3 = 10.
We can prove that it is the optimal answer.

Explaination2:

We can prove that it is the optimal answer.

Explaination2:

We will use shop1 and shop2.
We can prove that it is the optimal answer.

Mask Updates

Bitmasks are cool. A bitmask is a string of binary bits (0s and 1s). For example: "01010" is a bitmask.

Kuldeep is a naughty but brilliant computer scientist. In his free time, he writes some random programs to play with bitmasks. He has a PhD student under him and to test him (and entertain himself), he has given him the following task:

Given a number **N**, write a bitmask of length N containing all 0s. Now, you are given Q operations. Each operation contains two numbers (l, r) as input. An operation can be one of the following:

1. **Update** operation: Take the XOR of all the bits in the bitmask from index l to r (both inclusive) with 1.
2. **Query** operation: Count the number of set bits in the bitmask between index l to r (both inclusive).

You need to find the sum of all the queries.

Note:

1. In case there are no queries, return 0.
2. As the answer can be large, output the answer mod 1000000007
3. Consider 0 based indexing

Constraints:

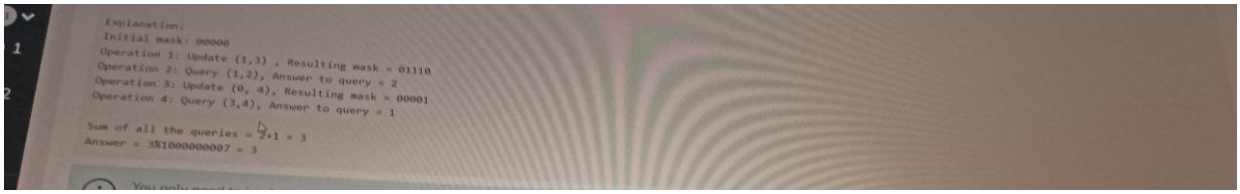
1 <= N <= 100000
1 <= Q <= 100000
0 <= l, r < N

Input format:

int A: Number N as described above
vector<vector<int> > B: Q operations as described above.
Each row contains three numbers (type, l, r). If type is 0, its an update operation else a query operation.

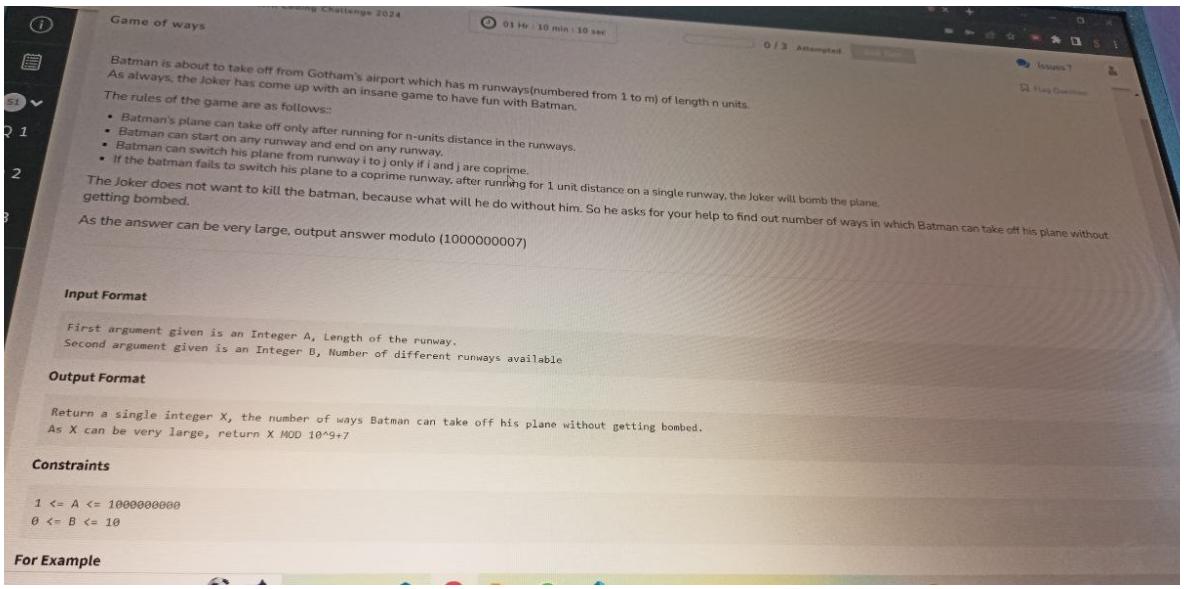
Example:

Input:



```
Explanation:
Initial mask: 00000
Operation 1: Update (1,3) , Resulting mask = 01110
Operation 2: Query (1,2), Answer to query = 2
Operation 3: Update (0, 4), Resulting mask = 00001
Operation 4: Query (3,4), Answer to query = 1

Sum of all the queries = 2+1 = 3
Answer = 3%1000000007 = 3
```



Game of ways

01 Hr : 10 min : 10 sec

0 / 3 Attempted

Batman is about to take off from Gotham's airport which has m runways(numbered from 1 to m) of length n units.
As always, the Joker has come up with an insane game to have fun with Batman.

The rules of the game are as follows::

- Batman's plane can take off only after running for n-units distance in the runways.
- Batman can start on any runway and end on any runway.
- Batman can switch his plane from runway i to j only if i and j are coprime.
- If the batman fails to switch his plane to a coprime runway, after running for 1 unit distance on a single runway, the Joker will bomb the plane.

The Joker does not want to kill the batman, because what will he do without him. So he asks for your help to find out number of ways in which Batman can take off his plane without getting bombed.

As the answer can be very large, output answer modulo (1000000007)

Input Format

First argument given is an Integer A, Length of the runway.
Second argument given is an Integer B, Number of different runways available

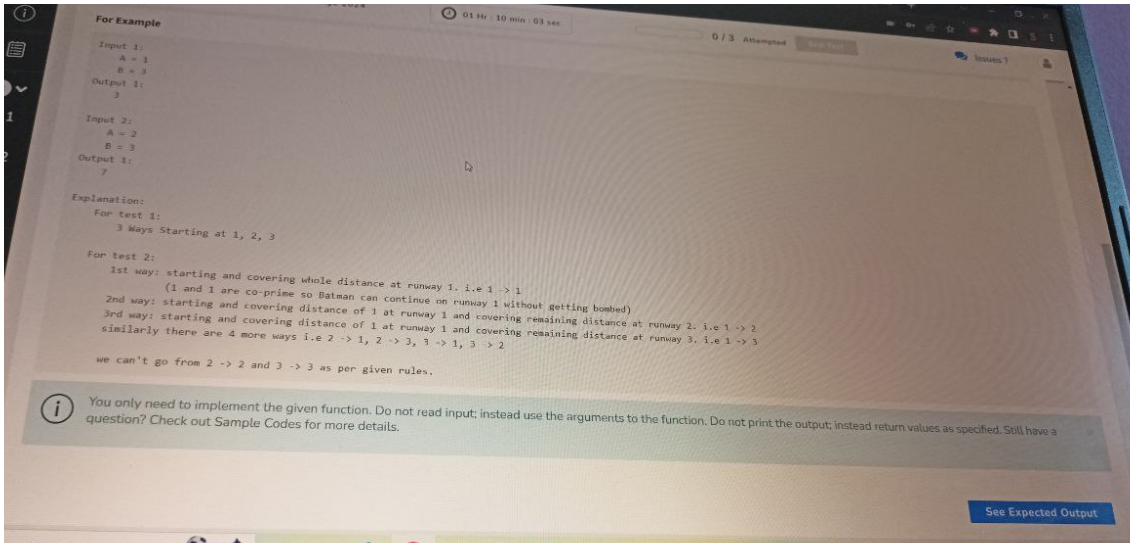
Output Format

Return a single integer X, the number of ways Batman can take off his plane without getting bombed.
As X can be very large, return X MOD 10^9+7

Constraints

```
1 <= A <= 1000000000
0 <= B <= 10
```

For Example



For Example

01 Hr : 10 min : 03 sec

0 / 3 Attempted

```
Input 1:
   A = 1
   B = 1
Output 1:
   3

Input 2:
   A = 2
   B = 3
Output 1:
   7

Explanation:
   For test 1:
      3 Ways Starting at 1, 2, 3

   For test 2:
      1st way: starting and covering whole distance at runway 1. i.e 1 -> 1
               (1 and 1 are co-prime so Batman can continue on runway 1 without getting bombed)
      2nd way: starting and covering distance of 1 at runway 1 and covering remaining distance at runway 2. i.e 1 -> 2
      3rd way: starting and covering distance of 1 at runway 1 and covering remaining distance at runway 3. i.e 1 -> 3
      similarly there are 4 more ways i.e 2 -> 1, 2 -> 3, 3 -> 1, 3 -> 2

      we can't go from 2 -> 2 and 3 -> 3 as per given rules.
```

You only need to implement the given function. Do not read input; instead use the arguments to the function. Do not print the output; instead return values as specified. Still have a question? Check out Sample Codes for more details.

See Expected Output