Consistent Salaries

Problem Statement

At a large bank, each of N employees e besides the CEO (employee #1) reports to exactly one person k_e (it is guaranteed that there are no cycles in the reporting graph). Initially, each employee e has salary s_e.

The CEO performs Q operations. Each operation i designates an employee e and a dollar amount v, and e and anyone who reports to e (directly or indirectly) has v added to their salary. Determine after which operations there exists a worker who makes strictly more than someone she reports to (directly or indirectly).

Input: The first line contains two space-separated integers N, Q. The second line contains $N-1$ space-separated integers $k_2, k_3, \ldots, k_N$ (employee 1 is the CEO and does not report to anyone). The third line contains N space-separated integers $s_1, s_2, \ldots, s_N$. Q lines follow, the i-th of which contains two space-separated integers e_i, v_i, representing the i-th operation.

Output: Q lines. The i-th line should be "GOOD" (without the quotes) if no worker has a strictly higher salary than someone she reports to and "BAD" otherwise.

Constraints: $1 \leq N, Q \leq 200000, 1 \leq k_e, e_i \leq N, 1 \leq s_e \leq 10^7, -1000 \leq v_i \leq 1000.$

Scoring:

- Test cases worth 30 points will have $1 \leq N, Q \leq 300.$
- Test cases worth an additional 20 points will have $1 \leq N, Q \leq 5000.$

Sample Input:

```
4 4
1 1 3
40 30 20 10
4 20
4 -20
2 15
1 30
```

Sample Output:

```
BAD
GOOD
BAD
BAD
```

Explanation: After the first operation, employee #4 makes more than his supervisor employee #3. After the second operation, no employee makes more than any of his supervisors. After the third operation, employee #2 makes more than his supervisor employee #1. After the fourth operation, employee #2 makes more than his supervisor employee #1.

Upstairs

Problem Statement

Joe goes to school in a very tall building. He attends N classes during the day, numbered chronologically from 1 through N. Class i is located on floor a_i. Joe's schedule allows him to switch the order of two consecutive classes. Since Joe does not like to walk up stairs, he wants to choose the schedule that requires him to walk up the least number of stairs throughout the day (each pair of consecutive floors is separated by the same number of stairs). Help Joe determine which classes to switch to minimize the number of stairs he must walk up during the school day, assuming he starts counting at the beginning of his first class.

Input: The first line of the input contains a single integer N. The second line of the input contains N space-separated integers $a_1, a_2, \ldots, a_N$.

Output: A single integer f, if it is optimal for Joe to switch classes f and $f+1$. If switching no classes is at least as good as the best possible switch, output -1. Otherwise, if there are multiple possible f, output the smallest one.

Constraints: $1 \leq N \leq 100000$, $1 \leq a_i \leq 1000$.

Scoring:

- Test cases worth a total of 70 points will have $N \leq 5000$.

Sample Input:

```
4
5 3 5 2
```

Sample Output:

```
2
```

Explanation: If classes 2 and 3 are switched, then Joe doesn't need to walk up any stairs during the day. This is clearly optimal.

[IOI] Skwishinese

Problem Statement

The language Skwishinese has its own elegant, albeit complicated, measure for the beauty of a word.

Before we can start describing how to measure it, we must first define a subsequence. Discovered by and named after the famous Skwishinese doctor, Aaron J. Subsequence, a *subsequence* of a word is another word that can be formed by removing one or more (possibly none) of the letters of that word. For example, `loooo` is a subsequence of `troolooloo`, but `trooooll` and `banana` are not.

Moreover, the *reverse* of a word is obtained simply by reversing the order of the letters. For example, the reverse of `troolooloo` is `oolooloort`. We denote the reverse of U by U^R.

Let U be a Skwishinese word. We can form a nonsensical word, UU^R, by *concatenating* U and its reverse. For example, if U is `loooo`, then UU^R will be `looooooool`.

Now, the *beauty* of a Skwishinese word V is the length of the shortest word U that is a subsequence of V and such that V is a subsequence of UU^R (the concatenation of U and its reverse).

For example, the beauty of the word `troolooloo` is 6, because 6 is the length of the shortest word U satisfying the above conditions (in this case, U is `troolo`).

Input Format
The input consists of exactly two lines.

The first line contains N, a positive integer giving the length of the word V.

The second line contains V a word in Skwishinese. It is a string consisting only of lowercase letters from the English alphabet.

Output Format
Output the beauty of the Skwishinese word V.

Constraints
$1 \leq N \leq 400$
V consists only of lowercase letters

Sample Input

```
6
oleleo
```

Sample Output

```
4
```

Explanation
One such possible U is `oele` (because `oele` is a subsequence of `oleleo`, and `oleleo` is a subsequence of UU^R = `oeleeleo`). Another is `olel`.

It can be shown that this is the shortest possible, so the answer is 4.

[IOI] Guardians of the Lunatics

HackerRank

Problem Statement

You are in charge of assigning guards to a prison where the craziest criminals are sent. The L cells form a single row and are numbered from 1 to L. Cell i houses exactly one lunatic whose *craziness level* is C_i.

Each lunatic should have one guard watching over him/her. Ideally, you should have one guard watching over each lunatic. However, due to budget constraints, you only have G guards to assign. You have to assign which lunatics each guard should watch over in order to minimize the total risk of having someone escape.

Of course, you should assign each guard to a set of adjacent cells. The *risk level* R_i that the lunatic in cell i can escape is given by the product of his/her craziness level C_i and the number of lunatics the guard assigned to him/her is watching over. Getting the sum of the R_i's from $i = 1$ to $i = L$ will give us the total amount of risk, R, that a lunatic might escape.

Given L lunatics and G guards, what is the minimum possible value of R?

Input Format
The first line of input contains two space-separated positive integers: L and G, the number of lunatics and the number of guards respectively.
The next L lines describe the craziness level of each of the lunatics. The i^{th} of these L lines describe the craziness level of the lunatic in cell block i.

Output Format
Output a line containing the minimum possible value of R.

Constraints
$1 \le L \le 8000$
$1 \le G \le 800$
$1 \le C_i \le 10^9$

Sample Input

```
6 3
11
11
11
24
26
100
```

Sample Output

```
299
```

Explanation
The first guard should be assigned to watch over the first three lunatics, each having a craziness level of 11. The second guard should be assigned to watch over the next two lunatics, having craziness levels of 24 and 26. The third guard should be assigned to the craziest lunatic, the one having a craziness level of 100.

The first three lunatics each have a risk level of 33, the product of 11 (their craziness level) and 3 (the number of lunatics their guard is watching over). The next three lunatics have risk level of 48, 52 and 100. Adding these up, the total risk level is 299.

[IOI] Christopher's Candy Chaos

Problem Statement

During its early years, Christopher's Candy Store offered only two flavors of candies - chocolate and strawberry. The Candy Store is popular for its "2 Chocolate 1 Strawberry" promo. You can buy a strip containing one strawberry candy and two chocolates for just the price of the chocolates! Talk about a steal!

Christopher has a machine that packs candies into capsules. These were known as candy strips. He just enters what flavors goes into a strip and the machine does it. But one day, the machine just malfunctioned and packed the candies randomly into one big strip. But that day, a high demand for the promo was met. And Christopher went into a panic.

Now, Christopher's idea is this: instead of throwing away the big strip and start anew, he should just trim the strip so that for every strawberry candy, there's two chocolate candies in the strip.

For example, if the strip was `SSCCSCSCC`, he could trim it from the left by removing the two strawberry candies. Then, trim it from the right and remove one chocolate candy. Thus, he's left with a strip looking like `CCSCSC`. This strip has one strawberry candy for every two chocolate candies.

As he recalls the fond memories of his newly-opened shop, he wondered how many ways he could trim each strip so that he could sell the remainder under the "2 Chocolate 1 Strawberry" promo.

Input Format
The first line contains a single integer, which is N.

The second line contains a string of length N, consisting of `S` and `C`.

Output Format
Output a single line containing how many ways he can tri the strips in order to make it sellable under the "2 Chocolate 1 Strawberry" promo.

Constraints
$1 \leq N \leq 10^6$

Sample Input

```
10
CCCCSSCCCC
```

Sample Output

```
7
```

Explanation
The seven possible trimmings are:

- `xxCCSxxxxx`
- `xxxxxSCCxx`
- `CCCCSSxxxx`

- `xCCCSSCxxx`
- `xxCCSSCCxx`
- `xxxCSSCCCx`
- `xxxxSSCCCC`

[IOI] World Peace

Problem Statement

The Artifact of World Peace was obtained by the evil Pharaoh. He smashed it into pieces and hid them inside a temple. The temple has a very elaborate and confusing floor plan. It contained different rooms. Each room has a trap. Moreover, to go from one room to another, you must pass by a corridor and each of these also has a trap. The Pharaoh suddenly realized he was low on budget because the college tuition fee of his children suddenly increased. Unprepared to sacrifice the future of his children, he decided to not push through with the traps inside the rooms BUT retain the traps in the corridors. Don't be fooled though, it is still very difficult to infiltrate this temple.

You decide that you want to get the pieces of this Sacred Artifact of *World Peace*, put these together to recreate the Artifact of *World Peace* and then finally obtain millions of dollars by selling it on eBay.

You list down all the known rooms of this temple. Each corridor has a trap and a certain skill level is required to clear this trap. Furthermore, to secure all pieces of the artifact, you realize that you just need to form the safest routes between pairs of rooms.

To summarize, the temple has R known rooms and C corridors. Each corridor connects two rooms and has its corresponding trap. Each trap has a certain *skill level* s required in order to clear it. If your skill level is below this s, you fail the trap and subsequently die a slow, horrible and painful death. After planning, you realize you just have to travel between P pairs of rooms. For each pair of rooms A and B, you must select the safest corridors (i.e. the path which requires the lowest skill level) to travel from room A to room B. Given these information, you must find out the minimum skill level you must have in order to survive travelling through the corridors which connect each of these pairs of rooms.

Input Format

The first line of input contains two integers, the number of rooms R in the temple and the number of corridors C in the temple. The rooms are numbered from 1 to R.

The next C lines each contain three integers: A, B and s in that order. This tells us that to survive the corridor connecting room A and room B, your skill level must be s or higher.

The next line contains a single integer, P. This indicates how many pairs of rooms you must plan a route for.

The next P lines each contain two integers, A and B. This tells us that you must plan the safest route between rooms A and B.

Output Format

Output a single line containing a single integer, S, the minimum skill level you must have in order to travel between each of the P pairs of rooms.

If it is impossible to travel between one of the P pairs of rooms, output `MISSION IMPOSSIBLE`.

Constraints

$1 \leq R \leq 10^5$
$1 \leq C \leq 10^5$
$1 \leq P \leq 10^5$
$1 \leq A, B \leq R$
$A \neq B$
$1 \leq s \leq 10^6$

Sample Input 1

```
8 12
1 2 1
1 4 2
2 3 5
2 4 4
3 4 2
5 2 11
5 2 10
5 6 1
6 8 3
7 6 2
8 7 3
5 8 4
2
1 3
5 8
```

Sample Output 1

```
3
```

Sample Input 2

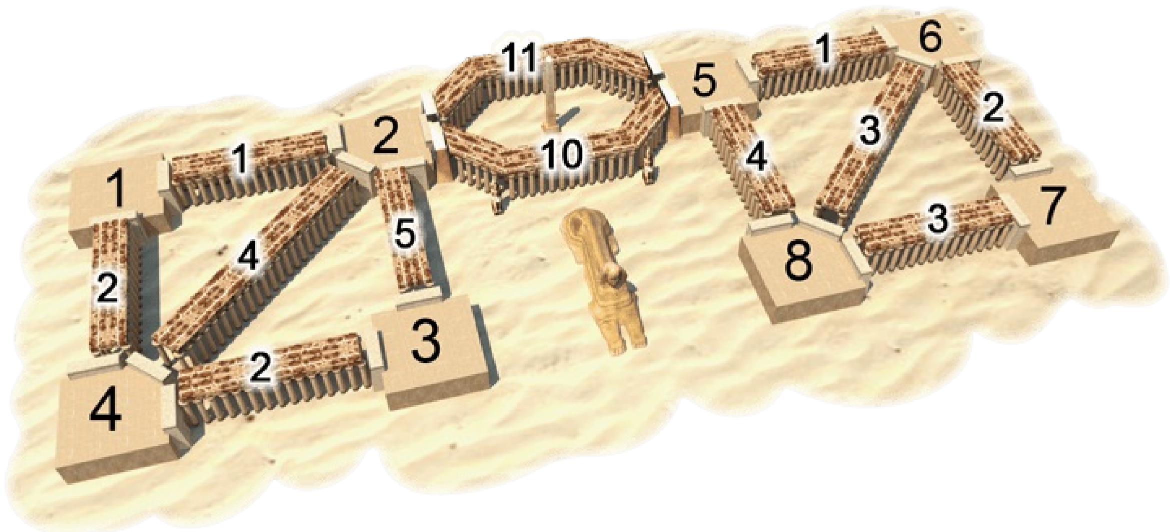
```
3 1
1 3 100
1
1 2
```

Sample Output 2

```
MISSION IMPOSSIBLE
```

Explanation

The first case corresponds to the following floor plan:



The safest path to travel from room 1 to room 3 is going from room 1 to room 4 then room 4 to room 3. The minimum skill level to safely traverse the corridors connecting these rooms is 2. The safest path to travel from room 5 to room 8 is going from room 5 to room 6, then room 7 and finally room 8 (or if you're feeling less adventurous, you could skip going to room 7 and travel from 5 to 6 to 8 instead). The minimum skill level to safely traverse these corridors is 3. Thus to travel between all $P = 2$ pairs of rooms, one must have a minimum skill level of 3.

For the second case, there is no way to get from room 1 to room 2 regardless of your skill level. How did they get the pieces there? It shall remain a mystery.

[IOI] Color Grid

HackerRank

Problem Statement

You are given an $N \times N$ grid. Each cell has the color white (color 0) in the beginning.

Each row and column has a certain color associated with it. *Filling a row or column* with a new color V means changing all the cells of that row or column to V (thus overriding the previous colors of the cells).

Now, given a sequence of P such operations, calculate the sum of the colors in the final grid.

For simplicity, the colors will be positive integers whose values will be most 10^9.

Input Format

The first line of input contains two integers N and P separated by a space.
The next P lines each contain a filling operation. There are two types of filling operations.

- `ROW I V` which means "fill row I with color V".
- `COL I V` which means "fill column I with color V".

Output Format

Output one line containing exactly one integer which is the sum of the colors in the final grid.

Constraints

$1 \leq N \leq 6000$
$1 \leq P \leq 400000$
$1 \leq I \leq N$
$1 \leq V \leq 10^9$

Sample Input

```
5 4
COL 1 6
COL 4 11
ROW 3 9
COL 1 24
```

Sample Output

```
200
```

Explanation

There are four operations. After the second operation, the grid looks like

```
6  0  0 11  0
6  0  0 11  0
6  0  0 11  0
6  0  0 11  0
6  0  0 11  0
```

After the third operation (`ROW 3 9`), the third row was colored with 9, overriding any previous color in the cells.

```
6  0  0 11  0
```

```
6  0  0 11  0
9  9  9  9  9
6  0  0 11  0
6  0  0 11  0
```

After the fourth operation (`COL 1 24`), the grid becomes:

```
24  0  0 11  0
24  0  0 11  0
24  9  9  9  9
24  0  0 11  0
24  0  0 11  0
```

The sum of the colors in this grid is 200.

[IOI] The Questions I Ask Myself

Problem Statement

Starting out with an empty sheet of paper, I start listing down random integers in increasing order. Every now and then, I stop to think. I look at part of my list and ask myself: "How many of these integers have M as a divisor?" I realized that I waste a lot of time doing this repeatedly and I wish to automate the process.

Input Format

The first line of input contains Q, the number of queries.

The following Q lines contain the queries. Each query is either `ADD V` or `ASK A B M` on a single line.

- `ADD V`. This adds the integer V at the end of our list. It is guaranteed that integers are added in increasing order.
- `ASK A B M`. For every query of this type, you need to output the number of integers V currently in our list such that V is at least A, at most B and has M as its divisor.

Output Format

For each `ASK` query, output a line containing the number of integers V currently on the list such that V is at least A, at most B and has M as its divisor.

Constraints

$1 \leq Q \leq 3 \cdot 10^5$
$1 \leq A \leq B \leq 3 \cdot 10^5$
$1 \leq M \leq 3 \cdot 10^5$
$1 \leq V \leq 3 \cdot 10^5$
The integers are added in increasing order.

Sample Input

```
10
ADD 1
ADD 2
ADD 3
ADD 5
ASK 2 8 2
ADD 8
ASK 2 8 2
ADD 13
ADD 21
ASK 1 300000 9
```

Sample Output

```
1
2
0
```

Explanation

During the first `ASK` query, the list contains `[1, 2, 3, 5]`. There is only **one** element in this list which is at least $A = 2$, at most $B = 8$ and has 2 as its factor - this is the element 2.

During the second `ASK` query, the list contains `[1, 2, 3, 5, 8]`. There are only **two** elements in this list which are at least $A = 2$, at most $B = 8$ and have 2 as its factor - these are the elements 2 and 8.

During the third `ASK` query, the list contains `[1, 2, 3, 5, 8, 13, 21]`. There are **no** elements in this list which are at least $A = 1$, at most $B = 300000$ and have 9 as its factor.

Guessing Numbers

Problem Statement

Joe picks an integer from the list $1, 2, \ldots, N$, with a probability p_i of picking i for all $1 \leq i \leq N$. He then gives Jason K attempts to guess his number. On each guess, Joe will tell Jason if his number is higher or lower than Jason's guess. If Jason guesses Joe's number correctly on any of the K guesses, the game terminates and Jason wins. Jason loses otherwise. If Jason knows all p_i's and plays optimally, what is the probability he wins?

Input: The first line contains two space-separated integers N, K. The second line contains N space-separated numbers $p_1, p_2, \ldots, p_N$, expressed as real numbers with 8 digits after the decimal point.

Output: A single line containing the probability that Jason wins. The output will be considered correct if its absolute error does not exceed 10^{-6}.

Constraints: $1 \leq N \leq 200000, 1 \leq K \leq 20, p_1 + p_2 + \cdots + p_n = 1, p_i \geq 0$.

Scoring:

- Test cases worth 20 points will have $N \leq 40$.
- Test cases worth an additional 40 points will have $N \leq 150$.

Sample Input:

```
5 1
0.20000000 0.30000000 0.40000000 0.10000000 0.00000000
```

Sample Output:

```
0.40000000
```

Explanation: With only one attempt, Jason wins iff his first guess is correct. So to maximize the probability he wins, he guesses 3 and wins with probability $p_3 = 0.4$.