Codeforces Round #379 (Div. 2)

A. Anton and Danik

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

Anton likes to play chess, and so does his friend Danik.

Once they have played n games in a row. For each game it's known who was the winner — Anton or Danik. None of the games ended with a tie.

Now Anton wonders, who won more games, he or Danik? Help him determine this.

Input

The first line of the input contains a single integer n $(1 \leq n \leq 100\,000)$ — the number of games played.

The second line contains a string s, consisting of n uppercase English letters 'A' and 'D' — the outcome of each of the games. The i-th character of the string is equal to 'A' if the Anton won the i-th game and 'D' if Danik won the i-th game.

Output

If Anton won more games than Danik, print "Anton" (without quotes) in the only line of the output.

If Danik won more games than Anton, print "Danik" (without quotes) in the only line of the output.

If Anton and Danik won the same number of games, print "Friendship" (without quotes).

Examples

input
6 ADAAAA
output
Anton

input
7 DDDAADA
output
Danik

input
6 DADADA
output
Friendship

Note

In the first sample, Anton won 6 games, while Danik — only 1. Hence, the answer is "Anton".

In the second sample, Anton won 3 games and Danik won 4 games, so the answer is "Danik".

In the third sample, both Anton and Danik won 3 games and the answer is "Friendship".

B. Anton and Digits

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

Recently Anton found a box with digits in his room. There are k_2 digits 2, k_3 digits 3, k_5 digits 5 and k_6 digits 6.

Anton's favorite integers are 32 and 256. He decided to compose this integers from digits he has. He wants to make the sum of these integers as large as possible. Help him solve this task!

Each digit can be used no more than once, i.e. the composed integers should contain no more than k_2 digits 2, k_3 digits 3 and so on. Of course, unused digits are not counted in the sum.

Input

The only line of the input contains four integers k_2, k_3, k_5 and k_6 — the number of digits 2, 3, 5 and 6 respectively ($0 \le k_2, k_3, k_5, k_6 \le 5 \cdot 10^6$).

Output

Print one integer — maximum possible sum of Anton's favorite integers that can be composed using digits from the box.

Examples

input
5 1 3 4
output
800

input
1 1 1 1
output
256

Note

In the first sample, there are five digits 2, one digit 3, three digits 5 and four digits 6. Anton can compose three integers 256 and one integer 32 to achieve the value $256 + 256 + 256 + 32 = 800$. Note, that there is one unused integer 2 and one unused integer 6. They are not counted in the answer.

In the second sample, the optimal answer is to create on integer 256, thus the answer is 256.

C. Anton and Making Potions

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Anton is playing a very interesting computer game, but now he is stuck at one of the levels. To pass to the next level he has to prepare n potions.

Anton has a special kettle, that can prepare one potions in x seconds. Also, he knows spells of two types that can faster the process of preparing potions.

1. Spells of this type speed up the preparation time of one potion. There are m spells of this type, the i-th of them costs b_i manapoints and changes the preparation time of each potion to a_i instead of x.
2. Spells of this type immediately prepare some number of potions. There are k such spells, the i-th of them costs d_i manapoints and instantly create c_i potions.

Anton can use **no more than one** spell of the first type and **no more than one** spell of the second type, and the total number of manapoints spent should not exceed s. Consider that all spells are used instantly and right before Anton starts to prepare potions.

Anton wants to get to the next level as fast as possible, so he is interested in the minimum number of time he needs to spent in order to prepare at least n potions.

Input

The first line of the input contains three integers n, m, k ($1 \le n \le 2 \cdot 10^9$, $1 \le m, k \le 2 \cdot 10^5$) — the number of potions, Anton has to make, the number of spells of the first type and the number of spells of the second type.

The second line of the input contains two integers x and s ($2 \le x \le 2 \cdot 10^9$, $1 \le s \le 2 \cdot 10^9$) — the initial number of seconds required to prepare one potion and the number of manapoints Anton can use.

The third line contains m integers a_i ($1 \le a_i < x$) — the number of seconds it will take to prepare one potion if the i-th spell of the first type is used.

The fourth line contains m integers b_i ($1 \le b_i \le 2 \cdot 10^9$) — the number of manapoints to use the i-th spell of the first type.

There are k integers c_i ($1 \le c_i \le n$) in the fifth line — the number of potions that will be immediately created if the i-th spell of the second type is used. It's guaranteed that c_i are **not decreasing**, i.e. $c_i \le c_j$ if $i < j$.

The sixth line contains k integers d_i ($1 \le d_i \le 2 \cdot 10^9$) — the number of manapoints required to use the i-th spell of the second type. It's guaranteed that d_i are **not decreasing**, i.e. $d_i \le d_j$ if $i < j$.

Output

Print one integer — the minimum time one has to spent in order to prepare n potions.

Examples

input
20 3 2 10 99 2 4 3 20 10 40 4 15 10 80
output
20

input
20 3 2 10 99 2 4 3 200 100 400 4 15 100 800
output
200

Note

In the first sample, the optimum answer is to use the second spell of the first type that costs 10 manapoints. Thus, the preparation time of each potion changes to 4 seconds. Also, Anton should use the second spell of the second type to instantly prepare 15 potions spending 80 manapoints. The total number of manapoints used is $10 + 80 = 90$, and the preparation time is $4 \cdot 5 = 20$ seconds (15 potions were prepared instantly, and the remaining 5 will take 4 seconds each).

In the second sample, Anton can't use any of the spells, so he just prepares 20 potions, spending 10 seconds on each of them and the answer is $20 \cdot 10 = 200$.

D. Anton and Chess

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Anton likes to play chess. Also, he likes to do programming. That is why he decided to write the program that plays chess. However, he finds the game on 8 to 8 board to too simple, he uses an infinite one instead.

The first task he faced is to check whether the king is in check. Anton doesn't know how to implement this so he asks you to help.

Consider that an infinite chess board contains one white king and the number of black pieces. There are only rooks, bishops and queens, as the other pieces are not supported yet. The white king is said to be in check if at least one black piece can reach the cell with the king in one move.

Help Anton and write the program that for the given position determines whether the white king is in check.

Remainder, on how do chess pieces move:

- Bishop moves any number of cells diagonally, but it can't "leap" over the occupied cells.
- Rook moves any number of cells horizontally or vertically, but it also can't "leap" over the occupied cells.
- Queen is able to move any number of cells horizontally, vertically or diagonally, but it also can't "leap".

Input

The first line of the input contains a single integer n $(1 \leq n \leq 500\,000)$ — the number of black pieces.

The second line contains two integers x_0 and y_0 (- $10^9 \leq x_0, y_0 \leq 10^9$) — coordinates of the white king.

Then follow n lines, each of them contains a character and two integers x_i and y_i (- $10^9 \leq x_i, y_i \leq 10^9$) — type of the i-th piece and its position. Character 'B' stands for the bishop, 'R' for the rook and 'Q' for the queen. It's guaranteed that no two pieces occupy the same position.

Output

The only line of the output should contains "YES" (without quotes) if the white king is in check and "NO" (without quotes) otherwise.

Examples

input
2 4 2 R 1 1 B 1 5
output
YES

input
2 4 2 R 3 3 B 1 5
output
NO

Note

Picture for the first sample:

White king is in check, because the black bishop can reach the cell with the white king in one move. The answer is "YES".
Picture for the second sample:

Here bishop can't reach the cell with the white king, because his path is blocked by the rook, and the bishop cant "leap" over it. Rook can't reach the white king, because it can't move diagonally. Hence, the king is not in check and the answer is "NO".

E. Anton and Tree

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

Anton is growing a tree in his garden. In case you forgot, the tree is a connected acyclic undirected graph.

There are n vertices in the tree, each of them is painted black or white. Anton doesn't like multicolored trees, so he wants to change the tree such that all vertices have the same color (black or white).

To change the colors Anton can use only operations of one type. We denote it as $paint(v)$, where v is some vertex of the tree. This operation changes the color of all vertices u such that all vertices on the shortest path from v to u have the same color (including v and u). For example, consider the tree

and apply operation $paint(3)$ to get the following:

Anton is interested in the minimum number of operation he needs to perform in order to make the colors of all vertices equal.

Input

The first line of the input contains a single integer n $(1 \le n \le 200\,000)$ — the number of vertices in the tree.

The second line contains n integers $color_i$ $(0 \le color_i \le 1)$ — colors of the vertices. $color_i = 0$ means that the i-th vertex is initially painted white, while $color_i = 1$ means it's initially painted black.

Then follow $n - 1$ line, each of them contains a pair of integers u_i and v_i $(1 \le u_i, v_i \le n, u_i \ne v_i)$ — indices of vertices connected by the corresponding edge. It's guaranteed that all pairs (u_i, v_i) are distinct, i.e. there are no multiple edges.

Output

Print one integer — the minimum number of operations Anton has to apply in order to make all vertices of the tree black or all vertices of the tree white.

Examples

input
11 0 0 0 1 1 0 1 0 0 1 1 1 2 1 3 2 4 2 5 5 6 5 7 3 8 3 9 3 10 9 11
output
2

input
4 0 0 0 0 1 2 2 3 3 4
output
0

Note

In the first sample, the tree is the same as on the picture. If we first apply operation $paint(3)$ and then apply $paint(6)$, the tree will become completely black, so the answer is 2.

In the second sample, the tree is already white, so there is no need to apply any operations and the answer is 0.

F. Anton and School

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

Anton goes to school, his favorite lessons are arraystudying. He usually solves all the tasks pretty fast, but this time the teacher gave him a complicated one: given two arrays b and c of length n, find array a, such that:

where $a\ and\ b$ means bitwise AND, while $a\ or\ b$ means bitwise OR.

Usually Anton is good in arraystudying, but this problem is too hard, so Anton asks you to help.

Input

The first line of the input contains a single integers n $(1 \leq n \leq 200\,000)$ — the size of arrays b and c.

The second line contains n integers b_i $(0 \leq b_i \leq 10^9)$ — elements of the array b.

Third line contains n integers c_i $(0 \leq c_i \leq 10^9)$ — elements of the array c.

Output

If there is no solution, print -1.

Otherwise, the only line of the output should contain n non-negative integers a_i — elements of the array a. If there are multiple possible solutions, you may print any of them.

Examples

input
4 6 8 4 4 16 22 10 10
output
3 5 1 1

input
5 8 25 14 7 16 19 6 9 4 25
output
-1