Codeforces Round #402 (Div. 2)

A. Pupils Redistribution

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

In Berland each high school student is characterized by *academic performance* — integer value between 1 and 5.

In high school `0xFF` there are two groups of pupils: the group A and the group B. Each group consists of exactly n students. An academic performance of each student is known — integer value between 1 and 5.

The school director wants to redistribute students between groups so that each of the two groups has the same number of students whose academic performance is equal to 1, the same number of students whose academic performance is 2 and so on. In other words, the purpose of the school director is to change the composition of groups, so that for each value of academic performance the numbers of students in both groups are equal.

To achieve this, there is a plan to produce a series of exchanges of students between groups. During the single exchange the director selects one student from the class A and one student of class B. After that, they both change their groups.

Print the least number of exchanges, in order to achieve the desired equal numbers of students for each academic performance.

Input

The first line of the input contains integer number n $(1 \le n \le 100)$ — number of students in both groups.

The second line contains sequence of integer numbers $a_1, a_2, ..., a_n$ $(1 \le a_i \le 5)$, where a_i is academic performance of the i-th student of the group A.

The third line contains sequence of integer numbers $b_1, b_2, ..., b_n$ $(1 \le b_i \le 5)$, where b_i is academic performance of the i-th student of the group B.

Output

Print the required minimum number of exchanges or `-1`, if the desired distribution of students can not be obtained.

Examples

input
```
4
5 4 4 4
5 5 4 5
```
output
```
1
```

input
```
6
1 1 1 1 1 1
5 5 5 5 5 5
```
output
```
3
```

input
```
1
5
3
```
output
```
-1
```

input
```
9
3 2 5 5 2 3 3 3 2
4 1 4 1 1 2 4 4 1
```
output
```
4
```

B. Weird Rounding

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

Polycarp is crazy about round numbers. He especially likes the numbers divisible by 10^k.

In the given number of n Polycarp wants to remove the least number of digits to get a number that is divisible by 10^k. For example, if $k = 3$, in the number `30020` it is enough to delete a single digit (`2`). In this case, the result is `3000` that is divisible by $10^3 = 1000$.

Write a program that prints the minimum number of digits to be deleted from the given integer number n, so that the result is divisible by 10^k. The result should not start with the unnecessary leading zero (i.e., zero can start only the number `0`, which is required to be written as exactly one digit).

It is guaranteed that the answer exists.

Input

The only line of the input contains two integer numbers n and k ($0 \le n \le 2\,000\,000\,000$, $1 \le k \le 9$).

It is guaranteed that the answer exists. All numbers in the input are written in traditional notation of integers, that is, without any extra leading zeros.

Output

Print w — the required minimal number of digits to erase. After removing the appropriate w digits from the number n, the result should have a value that is divisible by 10^k. The result can start with digit `0` in the single case (the result is zero and written by exactly the only digit `0`).

Examples

input

```
30020 3
```

output

```
1
```

input

```
100 9
```

output

```
2
```

input

```
10203049 2
```

output

```
3
```

Note

In the example 2 you can remove two digits: 1 and any 0. The result is number 0 which is divisible by any number.

C. Dishonest Sellers

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

Igor found out discounts in a shop and decided to buy n items. Discounts at the store will last for a week and Igor knows about each item that its price now is a_i, and after a week of discounts its price will be b_i.

Not all of sellers are honest, so now some products could be more expensive than after a week of discounts.

Igor decided that buy **at least** k of items now, but wait with the rest of the week in order to save money as much as possible. Your task is to determine the minimum money that Igor can spend to buy all n items.

Input

In the first line there are two positive integer numbers n and k ($1 \leq n \leq 2 \cdot 10^5$, $0 \leq k \leq n$) — total number of items to buy and minimal number of items Igor wants to by right now.

The second line contains sequence of integers $a_1, a_2, ..., a_n$ ($1 \leq a_i \leq 10^4$) — prices of items during discounts (i.e. right now).

The third line contains sequence of integers $b_1, b_2, ..., b_n$ ($1 \leq b_i \leq 10^4$) — prices of items after discounts (i.e. after a week).

Output

Print the minimal amount of money Igor will spend to buy all n items. Remember, he should buy at least k items right now.

Examples

input

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

output

```
10
```

input

```
5 3
3 4 7 10 3
4 5 5 12 5
```

output

```
25
```

Note

In the first example Igor should buy item 3 paying 6. But items 1 and 2 he should buy after a week. He will pay 3 and 1 for them. So in total he will pay $6 + 3 + 1 = 10$.

In the second example Igor should buy right now items 1, 2, 4 and 5, paying for them 3, 4, 10 and 3, respectively. Item 3 he should buy after a week of discounts, he will pay 5 for it. In total he will spend $3 + 4 + 10 + 3 + 5 = 25$.

D. String Game

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Little Nastya has a hobby, she likes to remove some letters from word, to obtain another word. But it turns out to be pretty hard for her, because she is too young. Therefore, her brother Sergey always helps her.

Sergey gives Nastya the word t and wants to get the word p out of it. Nastya removes letters in a certain order (one after another, in this order strictly), which is specified by permutation of letters' indices of the word t: $a_1 \ldots a_{|t|}$. We denote the length of word x as $|x|$. Note that after removing one letter, the indices of other letters don't change. For example, if $t =$ "nastya" and $a = [4, 1, 5, 3, 2, 6]$ then removals make the following sequence of words "nastya" "nas~~t~~ya" "~~n~~as~~t~~ya" "~~n~~as~~ty~~a" "~~n~~a~~sty~~a" "~~nasty~~a" "~~nastya~~".

Sergey knows this permutation. His goal is to stop his sister at some point and continue removing by himself to get the word p. Since Nastya likes this activity, Sergey wants to stop her as late as possible. Your task is to determine, how many letters Nastya can remove before she will be stopped by Sergey.

It is guaranteed that the word p can be obtained by removing the letters from word t.

Input

The first and second lines of the input contain the words t and p, respectively. Words are composed of lowercase letters of the Latin alphabet ($1 \le |p| < |t| \le 200\,000$). It is guaranteed that the word p can be obtained by removing the letters from word t.

Next line contains a permutation $a_1, a_2, \ldots, a_{|t|}$ of letter indices that specifies the order in which Nastya removes letters of t ($1 \le a_i \le |t|$, all a_i are distinct).

Output

Print a single integer number, the maximum number of letters that Nastya can remove.

Examples

input
```
ababcba
abb
5 3 4 1 7 6 2
```
output
```
3
```

input
```
bbbabb
bb
1 6 3 4 2 5
```
output
```
4
```

Note

In the first sample test sequence of removing made by Nastya looks like this:

"ababcba" "abab~~c~~ba" "ab~~a~~b~~c~~ba" "ab~~abc~~ba"

Nastya can not continue, because it is impossible to get word "abb" from word "~~a~~b~~abc~~ba".

So, Nastya will remove only three letters.

E. Bitwise Formula

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Bob recently read about bitwise operations used in computers: AND, OR and XOR. He have studied their properties and invented a new game.

Initially, Bob chooses integer m, bit depth of the game, which means that all numbers in the game will consist of m bits. Then he asks Peter to choose some m-bit number. After that, Bob computes the values of n variables. Each variable is assigned either a constant m-bit number or result of bitwise operation. Operands of the operation may be either variables defined before, or the number, chosen by Peter. After that, Peter's score equals to the sum of all variable values.

Bob wants to know, what number Peter needs to choose to get the minimum possible score, and what number he needs to choose to get the maximum possible score. In both cases, if there are several ways to get the same score, find the minimum number, which he can choose.

Input

The first line contains two integers n and m, the number of variables and bit depth, respectively ($1 \le n \le 5000$; $1 \le m \le 1000$).

The following n lines contain descriptions of the variables. Each line describes exactly one variable. Description has the following format: name of a new variable, space, sign ":=", space, followed by one of:

1. Binary number of exactly m bits.
2. The first operand, space, bitwise operation ("AND", "OR" or "XOR"), space, the second operand. Each operand is either the name of variable defined before or symbol '?', indicating the number chosen by Peter.

Variable names are strings consisting of lowercase Latin letters with length at most 10. All variable names are different.

Output

In the first line output the minimum number that should be chosen by Peter, to make the sum of all variable values minimum possible, in the second line output the minimum number that should be chosen by Peter, to make the sum of all variable values maximum possible. Both numbers should be printed as m-bit binary numbers.

Examples

input

```
3 3
a := 101
b := 011
c := ? XOR b
```

output

```
011
100
```

input

```
5 1
a := 1
bb := 0
cx := ? OR a
d := ? XOR ?
e := d AND bb
```

output

```
0
0
```

Note

In the first sample if Peter chooses a number 011_2, then $a = 101_2, b = 011_2, c = 000_2$, the sum of their values is 8. If he chooses the number 100_2, then $a = 101_2, b = 011_2, c = 111_2$, the sum of their values is 15.

For the second test, the minimum and maximum sum of variables a, bb, cx, d and e is 2, and this sum doesn't depend on the number chosen by Peter, so the minimum Peter can choose is 0.

F. Peterson Polyglot

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Peterson loves to learn new languages, but his favorite hobby is making new ones. Language is a set of words, and word is a sequence of lowercase Latin letters.

Peterson makes new language every morning. It is difficult task to store the whole language, so Peterson have invented new data structure for storing his languages which is called *broom*. Broom is rooted tree with edges marked with letters. Initially broom is represented by the only vertex — the root of the broom. When Peterson wants to add new word to the language he stands at the root and processes the letters of new word one by one. Consider that Peterson stands at the vertex u. If there is an edge from u marked with current letter, Peterson goes through this edge. Otherwise Peterson adds new edge from u to the new vertex v, marks it with the current letter and goes through the new edge. Size of broom is the number of vertices in it.

In the evening after working day Peterson can't understand the language he made this morning. It is too difficult for bored Peterson and he tries to make it simpler. Simplification of the language is the process of erasing some letters from some words of this language. Formally, Peterson takes some positive integer p and erases p-th letter from all the words of this language having length at least p. Letters in words are indexed starting by 1. Peterson considers that simplification should change at least one word, i.e. there has to be at least one word of length at least p. Peterson tries to make his language as simple as possible, so he wants to choose p such that the size of the broom for his simplified language is as small as possible.

Peterson is pretty annoyed with this task so he asks you for help. Write a program to find the smallest possible size of the broom and integer p.

Input

The first line of input contains integer n ($2 \le n \le 3 \cdot 10^5$) — the size of the broom.

Next n - 1 lines describe the broom: i-th of them contains integers u_i, v_i and letter x_i — describing the edge from u_i to v_i marked with letter x_i.

Vertices are numbered from 1 to n. All x_i are lowercase latin letters. Vertex 1 is the root of the broom.

Edges describe correct broom which is made from Peterson's language.

Output

The first line of output should contain the minimum possible size of the broom after its simplification. The second line of output should contain integer p to choose. If there are several suitable p values, print the smallest one.

Examples

input

```
5
1 2 c
2 3 a
3 4 t
2 5 t
```

output

```
3
2
```

input

```
16
1 2 o
2 3 f
1 4 p
4 5 i
5 6 e
6 7 c
7 8 e
4 9 r
9 10 e
10 11 t
11 12 t
12 13 y
10 14 f
14 15 i
15 16 x
```

output

```
12
2
```

Note

Broom from the second sample test can be built using language "piece", "of", "pie", "pretty", "prefix". Its simplification with $p = 2$ obtains the language of words "pece", "o", "pe", "petty", "pefix". This language gives us the broom with minimum possible size.