A. Presents

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

The Hedgehog likes to give presents to his friend, but no less he likes to receive them.

Having received another present today, the Hedgehog suddenly understood that he has no place to put it as there was no room left on the special shelf in the cupboard. He will have to choose another shelf, but which one should he choose, how large should it be?

In order to get to know this, the Hedgehog asks you to write him a program that will count the estimated number of presents that he will receive during the following N days. Besides, he is guided by the principle:

- on each holiday day the Hedgehog will necessarily receive a present,
- he receives presents at least every K days (i.e., if he received a present on the i-th day, he will receive the next present no later than on the $i+K$-th day).

For the given N and K, as well as the list of holidays among the following N days count the minimal number of presents that could be given to the Hedgehog. The number of today's day is zero, and you should regard today's present as already given (i.e., you shouldn't count it in the answer).

Input

The first line contains integers N and K ($1 \leq N \leq 365$, $1 \leq K \leq N$).

The second line contains a number C which represents the number of holidays ($0 \leq C \leq N$). Then in the same line follow C numbers ranging from 1 to N which are the numbers of holiday days. The numbers are given in the increasing order, without repeating numbers among them.

Output

Print a single number — the minimal number of presents the Hedgehog will receive over the following N days.

Examples

input

```
5 2
1 3
```

output

```
3
```

input

```
10 1
3 6 7 8
```

output

```
10
```

B. Cutting Jigsaw Puzzle

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

The Hedgehog recently remembered one of his favorite childhood activities, — solving puzzles, and got into it with new vigor. He would sit day in, day out with his friend buried into thousands of tiny pieces of the picture, looking for the required items one by one.

Soon the Hedgehog came up with a brilliant idea: instead of buying ready-made puzzles, one can take his own large piece of paper with some picture and cut it into many small rectangular pieces, then mix them and solve the resulting puzzle, trying to piece together the picture. The resulting task is even more challenging than the classic puzzle: now all the fragments have the same rectangular shape, and one can assemble the puzzle only relying on the picture drawn on the pieces.

All puzzle pieces turn out to be of the same size $X \times Y$, because the picture is cut first by horizontal cuts with the pitch of X, then with vertical cuts with the pitch of Y. If we denote the initial size of the picture as $A \times B$, then A must be divisible by X and B must be divisible by Y (X and Y are integer numbers).

However, not every such cutting of the picture will result in a **good** puzzle. The Hedgehog finds a puzzle good if no two pieces in it are the same (It is **allowed to rotate** the pieces when comparing them, but it is forbidden to turn them over).

Your task is to count for a given picture the number of good puzzles that you can make from it, and also to find the puzzle with the minimal piece size.

Input

The first line contains two numbers A and B which are the sizes of the picture. They are positive integers not exceeding 20.

Then follow A lines containing B symbols each, describing the actual picture. The lines only contain uppercase English letters.

Output

In the first line print the number of possible good puzzles (in other words, the number of pairs (X, Y) such that the puzzle with the corresponding element sizes will be good). This number should always be positive, because the whole picture is a good puzzle itself.

In the second line print two numbers — the sizes X and Y of the smallest possible element among all good puzzles. The comparison is made firstly by the area XY of one element and secondly — by the length X.

Examples

input

```
2 4
ABDC
ABDC
```

output

```
3
2 1
```

input

```
2 6
ABCCBA
ABCCBA
```

output

```
1
2 6
```

Note

The picture in the first sample test has the following good puzzles: $(2, 1)$, $(2, 2)$, $(2, 4)$.

C. First Digit Law

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

In the probability theory the following paradox called Benford's law is known: "In many lists of random numbers taken from real sources, numbers starting with digit 1 occur much more often than numbers starting with any other digit" (that's the simplest form of the law).

Having read about it on Codeforces, the Hedgehog got intrigued by the statement and wishes to thoroughly explore it. He finds the following similar problem interesting in particular: there are N random variables, the i-th of which can take any integer value from some segment $[L_i;R_i]$ (all numbers from this segment are equiprobable). It means that the value of the i-th quantity can be equal to any integer number from a given interval $[L_i;R_i]$ with probability $1 / (R_i - L_i + 1)$.

The Hedgehog wants to know the probability of the event that the first digits of at least $K\%$ of those values will be equal to one. In other words, let us consider some set of fixed values of these random variables and leave only the first digit (the MSD — most significant digit) of each value. Then let's count how many times the digit 1 is encountered and if it is encountered in at least K per cent of those N values, than such set of values will be called a good one. You have to find the probability that a set of values of the given random variables will be a good one.

Input

The first line contains number N which is the number of random variables ($1 \le N \le 1000$). Then follow N lines containing pairs of numbers L_i, R_i, each of whom is a description of a random variable. It is guaranteed that $1 \le L_i \le R_i \le 10^{18}$.

The last line contains an integer K ($0 \le K \le 100$).

All the numbers in the input file are integers.

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cin` (also you may use `%I64d`).

Output

Print the required probability. Print the fractional number with such a precision that the relative or absolute error of the result won't exceed 10^{-9}.

Examples

input

```
1
1 2
50
```

output

```
0.500000000000000
```

input

```
2
1 2
9 11
50
```

output

```
0.833333333333333
```

D. Writing a Song

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

One of the Hedgehog and his friend's favorite entertainments is to take some sentence or a song and replace half of the words (sometimes even all of them) with each other's names.

The friend's birthday is approaching and the Hedgehog decided to make a special present to his friend: a very long song, where his name will be repeated many times. But try as he might, he can't write a decent song!

The problem is that the Hedgehog has already decided how long the resulting sentence should be (i.e. how many letters it should contain) and in which positions in the sentence the friend's name should occur, and it must not occur in any other position in the sentence. Besides, the Hedgehog decided to limit himself to using only the first K letters of an English alphabet in this sentence (so it will be not even a sentence, but one long word).

The resulting problem is indeed quite complicated, that's why the Hedgehog asks you to help him and write a program that will make the desired word by the given name P, the length N of the required word, the given positions of the occurrences of the name P in the desired word and the alphabet's size K. Note that the occurrences of the name can overlap with each other.

Input

The first line contains numbers N and K which are the length of the required string and the alphabet size accordingly. The limitations are: $1 \le N \le 100$, $2 \le K \le 26$.

The second line contains the name P which is a non-empty string whose length does not exceed N characters. The string consists only of the first K lowercase symbols of an English alphabet.

The third line contains the string of length $N - length(P) + 1$, consisting only of numbers zero and one. A number one in the i-th position means that an occurrence of the name P should start from i-th position of the desired word, while a zero means that there is no occurrence starting here.

Output

Print the desired word S. If there are several answers, print any of them.

If there is no solution, then print "No solution".

Examples

input

```
5 2
aba
101
```

output

```
ababa
```

input

```
5 2
a
10001
```

output

```
abbba
```

input

```
6 2
abba
101
```

output

```
No solution
```

E. Vacuum Cleaner

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

One winter evening the Hedgehog was relaxing at home in his cozy armchair and clicking through the TV channels. Stumbled on an issue of «TopShop», the Hedgehog was about to change the channel when all of a sudden he was stopped by an advertisement of a new wondrous invention.

Actually, a vacuum cleaner was advertised there. It was called Marvellous Vacuum and it doesn't even need a human to operate it while it cleans! The vacuum cleaner can move around the flat on its own: it moves in some direction and if it hits an obstacle there, it automatically chooses a new direction. Sooner or later this vacuum cleaner will travel through all the room and clean it all. Having remembered how much time the Hedgehog spends every time on cleaning (surely, no less than a half of the day), he got eager to buy this wonder.

However, the Hedgehog quickly understood that the cleaner has at least one weak point: it won't clean well in the room's corners because it often won't able to reach the corner due to its shape. To estimate how serious is this drawback in practice, the Hedgehog asked you to write for him the corresponding program.

You will be given the cleaner's shape in the top view. We will consider only the cases when the vacuum cleaner is represented as a convex polygon. The room is some infinitely large rectangle. We consider one corner of this room and want to find such a rotation of the vacuum cleaner so that it, being pushed into this corner, will leave the minimum possible area in the corner uncovered.

Input

The first line contains an integer N which represents the number of vertices of the vacuum cleaner's polygon ($3 \le N \le 4 \cdot 10^4$). Then follow N lines each containing two numbers — the coordinates of a vertex of the polygon. All the coordinates are integer and their absolute values do not exceed 10^6.

It is guaranteed that the given polygon is nondegenerate and convex (no three points lie on the same line). The polygon vertices are given in a clockwise or counter-clockwise direction.

Output

Print the minimum possible uncovered area. The answer will be accepted if it is within 10^{-6} of absolute or relative error from the correct answer.

Examples

input

```
4
0 0
1 0
1 1
0 1
```

output

```
0.00000000000000000000
```

input

```
8
1 2
2 1
2 -1
1 -2
-1 -2
-2 -1
-2 1
-1 2
```

output

```
0.50000000000000000000
```