Codeforces Round #233 (Div. 2)

A. Pages

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

User ainta is making a web site. This time he is going to make a navigation of the pages. In his site, there are n pages numbered by integers from 1 to n. Assume that somebody is on the p-th page now. The navigation will look like this:

$$<<\ p-k\ p-k+1\ \ ...\ \ p-1\ \ (p)\ \ p+1\ \ ...\ \ p+k-1\ \ p+k\ >>$$

When someone clicks the button "<<" he is redirected to page 1, and when someone clicks the button ">>" he is redirected to page n. Of course if someone clicks on a number, he is redirected to the corresponding page.

There are some conditions in the navigation:

- If page 1 is in the navigation, the button "<<" must not be printed.
- If page n is in the navigation, the button ">>" must not be printed.
- If the page number is smaller than 1 or greater than n, it must not be printed.

You can see some examples of the navigations. Make a program that prints the navigation.

Input

The first and the only line contains three integers n, p, k ($3 \le n \le 100$; $1 \le p \le n$; $1 \le k \le n$)

Output

Print the proper navigation. Follow the format of the output from the test samples.

Sample test(s)

input
17 5 2
output
<< 3 4 (5) 6 7 >>

input
6 5 2
output
<< 3 4 (5) 6

input
6 1 2
output
(1) 2 3 >>

input
6 2 2
output
1 (2) 3 4 >>

input
9 6 3
output
<< 3 4 5 (6) 7 8 9

input
10 6 3
output
<< 3 4 5 (6) 7 8 9 >>

input
8 5 4
output
1 2 3 4 (5) 6 7 8

B. Red and Blue Balls

time limit per test: 0.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

User ainta has a stack of n red and blue balls. He can apply a certain operation which changes the colors of the balls inside the stack.

- While the top ball inside the stack is red, pop the ball from the top of the stack.
- Then replace the blue ball on the top with a red ball.
- And finally push some blue balls to the stack until the stack has total of n balls inside.

If there are no blue balls inside the stack, ainta can't apply this operation. Given the initial state of the stack, ainta wants to know the maximum number of operations he can repeatedly apply.

Input

The first line contains an integer n $(1 \le n \le 50)$ — the number of balls inside the stack.

The second line contains a string s ($|s| = n$) describing the initial state of the stack. The i-th character of the string s denotes the color of the i-th ball (we'll number the balls from top to bottom of the stack). If the character is "R", the color is red. If the character is "B", the color is blue.

Output

Print the maximum number of operations ainta can repeatedly apply.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
3 RBR
output
2

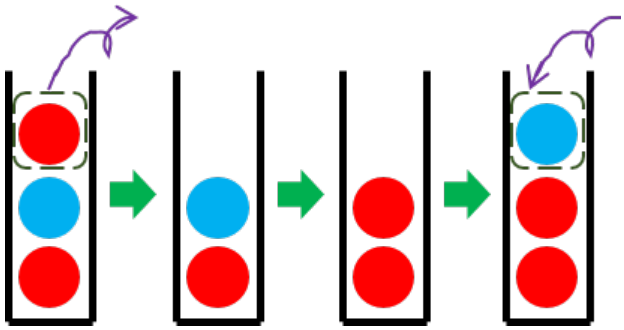
input
4 RBBR
output
6

input
5 RBBRR
output
6

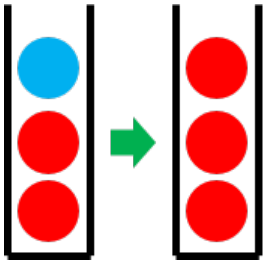
Note

The first example is depicted below.

The explanation how user ainta applies the first operation. He pops out one red ball, changes the color of the ball in the middle from blue to red, and pushes one blue ball.

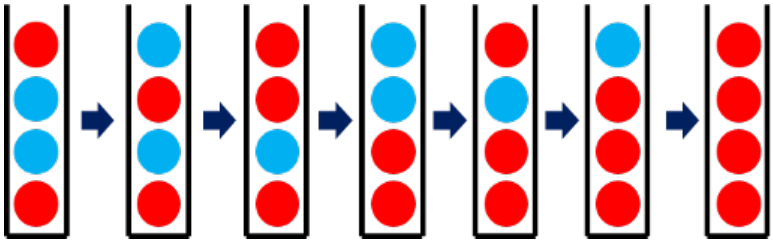


The explanation how user ainta applies the second operation. He will not pop out red balls, he simply changes the color of the ball on the top from blue to red.



From now on, ainta can't apply any operation because there are no blue balls inside the stack. ainta applied two operations, so the answer is 2.

The second example is depicted below. The blue arrow denotes a single operation.



C. Cards

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

User ainta loves to play with cards. He has a cards containing letter "o" and b cards containing letter "x". He arranges the cards in a row, and calculates the score of the deck by the formula below.

1. At first, the score is 0.
2. For each block of contiguous "o"s with length x the score increases by x^2.
3. For each block of contiguous "x"s with length y the score decreases by y^2.

For example, if $a = 6, b = 3$ and ainta have arranged the cards in the order, that is described by string "ooxoooxxo", the score of the deck equals $2^2 - 1^2 + 3^2 - 2^2 + 1^2 = 9$. That is because the deck has 5 blocks in total: "oo", "x", "ooo", "xx", "o".

User ainta likes big numbers, so he wants to maximize the score with the given cards. Help ainta make the score as big as possible. Note, that he has to arrange all his cards.

Input

The first line contains two space-separated integers a and b ($0 \le a, b \le 10^5$; $a + b \ge 1$) — the number of "o" cards and the number of "x" cards.

Output

In the first line print a single integer v — the maximum score that ainta can obtain.

In the second line print $a + b$ characters describing the deck. If the k-th card of the deck contains "o", the k-th character must be "o". If the k-th card of the deck contains "x", the k-th character must be "x". The number of "o" characters must be equal to a, and the number of "x " characters must be equal to b. If there are many ways to maximize v, print any.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
2 3
output
-1 xoxox

input
4 0
output
16 oooo

input
0 4
output
-16 xxxx

D. Painting The Wall

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

User ainta decided to paint a wall. The wall consists of n^2 tiles, that are arranged in an $n \times n$ table. Some tiles are painted, and the others are not. As he wants to paint it beautifully, he will follow the rules below.

1. Firstly user ainta looks at the wall. If there is at least one painted cell on each row and at least one painted cell on each column, he stops coloring. Otherwise, he goes to step 2.
2. User ainta choose any tile on the wall with uniform probability.
3. If the tile he has chosen is not painted, he paints the tile. Otherwise, he ignores it.
4. Then he takes a rest for one minute even if he doesn't paint the tile. And then ainta goes to step 1.

However ainta is worried if it would take too much time to finish this work. So he wants to calculate the expected time needed to paint the wall by the method above. Help him find the expected time. You can assume that choosing and painting any tile consumes no time at all.

Input

The first line contains two integers n and m ($1 \le n \le 2 \cdot 10^3; 0 \le m \le min(n^2, 2 \cdot 10^4)$) — the size of the wall and the number of painted cells.

Next m lines goes, each contains two integers r_i and c_i ($1 \le r_i, c_i \le n$) — the position of the painted cell. It is guaranteed that the positions are all distinct. Consider the rows of the table are numbered from 1 to n. Consider the columns of the table are numbered from 1 to n.

Output

In a single line print the expected time to paint the wall in minutes. Your answer will be considered correct if it has at most 10^{-4} absolute or relative error.

Sample test(s)

input
5 2 2 3 4 1
output
11.7669491886

input
2 2 1 1 1 2
output
2.0000000000

input
1 1 1 1
output
0.0000000000

E. Tree and Array

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

User ainta likes trees. This time he is going to make an undirected tree with n vertices numbered by integers from 1 to n. The tree is weighted, so each edge of the tree will have some integer weight.

Also he has an array t: $t[1], t[2], ..., t[n]$. At first all the elements of the array are initialized to 0. Then for each edge connecting vertices u and v ($u < v$) of the tree with weight c, ainta adds value c to the elements $t[u], t[u + 1], ..., t[v - 1], t[v]$ of array t.

Let's assume that $d(u, v)$ is the total weight of edges on the shortest path between vertex u and vertex v. User ainta calls a pair of integers x, y ($1 \le x < y \le n$) *good* if and only if $d(x, y) = t[x] + t[x + 1] + ... + t[y - 1] + t[y]$.

User ainta wants to make at least $\lfloor \frac{n}{2} \rfloor$ good pairs, but he couldn't make a proper tree. Help ainta to find such a tree.

Input

The first line contains a single integer n ($5 \le n \le 10^5$).

Output

Print $n - 1$ lines containing the description of the edges. The i-th line should contain three space-separated integers u_i, v_i, c_i ($1 \le u_i < v_i \le n; 1 \le c_i \le 10^5$) — two vertices connected by the edge, and the weight of the edge.

Next print $\lfloor \frac{n}{2} \rfloor$ lines containing the good pairs. The k-th line should contain two space-separated integers x_k and y_k ($1 \le x_k < y_k \le n$). Of course, x_k, y_k must be a good pair. All pairs should be distinct — that is, for all $j, k (1 \le j < k \le \lfloor \frac{n}{2} \rfloor)$, $x_j \ne x_k$ or $y_j \ne y_k$ must be satisfied.

If there are many correct solutions, print any of them.

Sample test(s)

input
7
output
1 4 1 1 2 2 2 3 5 3 5 3 2 6 2 6 7 3 4 5 5 6 5 7

Note

$\lfloor x \rfloor$ is the largest integer not greater than x.

You can find the definition of a tree by the following link: `http://en.wikipedia.org/wiki/Tree_(graph_theory)`

You can also find the definition of the shortest path by the following link: `http://en.wikipedia.org/wiki/Shortest_path_problem`

The tree and the array t in the sample output look like this: