Rockethon 2015

A. Game

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

Two players play a simple game. Each player is provided with a box with balls. First player's box contains exactly n_1 balls and second player's box contains exactly n_2 balls. In one move first player can take from 1 to k_1 balls from his box and throw them away. Similarly, the second player can take from 1 to k_2 balls from his box in his move. Players alternate turns and the first player starts the game. The one who can't make a move loses. Your task is to determine who wins if both players play optimally.

Input

The first line contains four integers n_1, n_2, k_1, k_2. All numbers in the input are from 1 to 50.

This problem doesn't have subproblems. You will get 3 points for the correct submission.

Output

Output "First" if the first player wins and "Second" otherwise.

Sample test(s)

input
2 2 1 2
output
Second

input
2 1 1 1
output
First

Note

Consider the first sample test. Each player has a box with 2 balls. The first player draws a single ball from his box in one move and the second player can either take 1 or 2 balls from his box in one move. No matter how the first player acts, the second player can always win if he plays wisely.

B1. Permutations

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

You are given a permutation p of numbers $1, 2, ..., n$. Let's define $f(p)$ as the following sum:

$$f(p) = \sum_{i=1}^{n} \sum_{j=i}^{n} min(p_i, p_{i+1}, ...p_j)$$

Find the lexicographically m-th permutation of length n in the set of permutations having the maximum possible value of $f(p)$.

Input

The single line of input contains two integers n and m ($1 \le m \le cnt_n$), where cnt_n is the number of permutations of length n with maximum possible value of $f(p)$.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem B1 (3 points), the constraint $1 \le n \le 8$ will hold.
- In subproblem B2 (4 points), the constraint $1 \le n \le 50$ will hold.

Output

Output n number forming the required permutation.

Sample test(s)

input
2 2
output
2 1

input
3 2
output
1 3 2

Note

In the first example, both permutations of numbers {1, 2} yield maximum possible $f(p)$ which is equal to 4. Among them, $(2, 1)$ comes second in lexicographical order.

B2. Permutations

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

You are given a permutation p of numbers $1, 2, ..., n$. Let's define $f(p)$ as the following sum:

$$f(p) = \sum_{i=1}^{n} \sum_{j=i}^{n} min(p_i, p_{i+1}, ...p_j)$$

Find the lexicographically m-th permutation of length n in the set of permutations having the maximum possible value of $f(p)$.

Input

The single line of input contains two integers n and m ($1 \le m \le cnt_n$), where cnt_n is the number of permutations of length n with maximum possible value of $f(p)$.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem B1 (3 points), the constraint $1 \le n \le 8$ will hold.
- In subproblem B2 (4 points), the constraint $1 \le n \le 50$ will hold.

Output

Output n number forming the required permutation.

Sample test(s)

input
2 2
output
2 1

input
3 2
output
1 3 2

Note

In the first example, both permutations of numbers {1, 2} yield maximum possible $f(p)$ which is equal to 4. Among them, $(2, 1)$ comes second in lexicographical order.

C. Second price auction

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

Nowadays, most of the internet advertisements are not statically linked to a web page. Instead, what will be shown to the person opening a web page is determined within 100 milliseconds after the web page is opened. Usually, multiple companies compete for each ad slot on the web page in an auction. Each of them receives a request with details about the user, web page and ad slot and they have to respond within those 100 milliseconds with a bid they would pay for putting an advertisement on that ad slot. The company that suggests the highest bid wins the auction and gets to place its advertisement. If there are several companies tied for the highest bid, the winner gets picked at random.

However, the company that won the auction does not have to pay the exact amount of its bid. In most of the cases, a second-price auction is used. This means that the amount paid by the company is equal to the maximum of all the other bids placed for this ad slot.

Let's consider one such bidding. There are n companies competing for placing an ad. The i-th of these companies will bid an integer number of microdollars equiprobably randomly chosen from the range between L_i and R_i, inclusive. In the other words, the value of the i-th company bid can be any integer from the range $[L_i, R_i]$ with the same probability.

Determine the expected value that the winner will have to pay in a second-price auction.

Input

The first line of input contains an integer number n ($2 \le n \le 5$). n lines follow, the i-th of them containing two numbers L_i and R_i ($1 \le L_i \le R_i \le 10000$) describing the i-th company's bid preferences.

This problem doesn't have subproblems. You will get 8 points for the correct submission.

Output

Output the answer with absolute or relative error no more than $1e-9$.

Sample test(s)

input
3 4 7 8 10 5 5
output
5.7500000000

input
3 2 5 3 4 1 6
output
3.5000000000

Note

Consider the first example. The first company bids a random integer number of microdollars in range $[4, 7]$; the second company bids between 8 and 10, and the third company bids 5 microdollars. The second company will win regardless of the exact value it bids, however the price it will pay depends on the value of first company's bid. With probability 0.5 the first company will bid at most 5 microdollars, and the second-highest price of the whole auction will be 5. With probability 0.25 it will bid 6 microdollars, and with probability 0.25 it will bid 7 microdollars. Thus, the expected value the second company will have to pay is $0.5 \cdot 5 + 0.25 \cdot 6 + 0.25 \cdot 7 = 5.75$.

D1. Constrained Tree

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

You need to find a binary tree of size n that satisfies a given set of c constraints. Suppose that the nodes of the unknown binary tree are labeled using a *pre-order* traversal starting with 1. For the i-th constraint you are given two labels, a_i and b_i and a direction, left or right. In case of left direction, b_i is an element of the subtree rooted at a_i's left child. Similarly in the case of right direction b_i is an element of the subtree rooted at a_i's right child.

Input

The first line of input contains two integers n and c. The next c lines contain 2 integers a_i, b_i ($1 \le a_i, b_i \le n$) and either "LEFT" or "RIGHT" denoting whether b is in the subtree rooted at a_i's left child or in the subtree rooted at a_i's right child.

The problem consists of multiple subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem D1 (9 points), the constraints $1 \le n \le 100$, $1 \le c \le 50$ will hold.
- In subproblem D2 (8 points), the constraints $1 \le n \le 1000000$, $1 \le c \le 100000$ will hold.

Output

Output will be on a single line.

Any binary tree that satisfies the constraints will be accepted. The tree's nodes should be printed out as n space separated labels representing an *in-order* traversal, using the *pre-order* numbers as labels of vertices.

If there are no trees that satisfy the constraints, print "IMPOSSIBLE" (without quotes).

Sample test(s)

input
3 2 1 2 LEFT 1 3 RIGHT
output
2 1 3

input
3 2 1 2 RIGHT 1 3 LEFT
output
IMPOSSIBLE

Note

Consider the first sample test. We need to find a tree with 3 nodes that satisfies the following two constraints. The node labeled 2 with pre-order traversal should be in the left subtree of the node labeled 1 with pre-order traversal; the node labeled 3 with pre-order traversal should be in the right subtree of the node labeled 1. There is only one tree with three nodes that satisfies these constraints and its in-order traversal is $(2, 1, 3)$.

Pre-order is the "root – left subtree – right subtree" order. *In-order* is the "left subtree – root – right subtree" order.

For other information regarding *in-order* and *pre-order*, see http://en.wikipedia.org/wiki/Tree_traversal.

D2. Constrained Tree

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

You need to find a binary tree of size n that satisfies a given set of c constraints. Suppose that the nodes of the unknown binary tree are labeled using a *pre-order* traversal starting with 1. For the i-th constraint you are given two labels, a_i and b_i and a direction, left or right. In case of left direction, b_i is an element of the subtree rooted at a_i's left child. Similarly in the case of right direction b_i is an element of the subtree rooted at a_i's right child.

Input

The first line of input contains two integers n and c. The next c lines contain 2 integers a_i, b_i ($1 \le a_i, b_i \le n$) and either "LEFT" or "RIGHT" denoting whether b is in the subtree rooted at a_i's left child or in the subtree rooted at a_i's right child.

The problem consists of multiple subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem D1 (9 points), the constraints $1 \le n \le 100$, $1 \le c \le 50$ will hold.
- In subproblem D2 (8 points), the constraints $1 \le n \le 1000000$, $1 \le c \le 100000$ will hold.

Output

Output will be on a single line.

Any binary tree that satisfies the constraints will be accepted. The tree's nodes should be printed out as n space separated labels representing an *in-order* traversal, using the *pre-order* numbers as labels of vertices.

If there are no trees that satisfy the constraints, print "IMPOSSIBLE" (without quotes).

Sample test(s)

input
3 2 1 2 LEFT 1 3 RIGHT
output
2 1 3

input
3 2 1 2 RIGHT 1 3 LEFT
output
IMPOSSIBLE

Note

Consider the first sample test. We need to find a tree with 3 nodes that satisfies the following two constraints. The node labeled 2 with pre-order traversal should be in the left subtree of the node labeled 1 with pre-order traversal; the node labeled 3 with pre-order traversal should be in the right subtree of the node labeled 1. There is only one tree with three nodes that satisfies these constraints and its in-order traversal is $(2, 1, 3)$.

Pre-order is the "root – left subtree – right subtree" order. *In-order* is the "left subtree – root – right subtree" order.

For other information regarding *in-order* and *pre-order*, see http://en.wikipedia.org/wiki/Tree_traversal.

E1. Subarray Cuts

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

You are given an array of length n and a number k. Let's pick k non-overlapping non-empty subarrays of the initial array. Let s_i be the sum of the i-th subarray in order from left to right. Compute the maximum value of the following expression:

$$|s_1 - s_2| + |s_2 - s_3| + \ldots + |s_{k-1} - s_k|$$

Here subarray is a contiguous part of an array.

Input

The first line of input contains two integers n and k. The second line contains n integers — the elements of the array. The absolute values of elements do not exceed 10^4.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem E1 (9 points), constraints $2 \le n \le 400$, $2 \le k \le min(n, 50)$ will hold.
- In subproblem E2 (12 points), constraints $2 \le n \le 30000$, $2 \le k \le min(n, 200)$ will hold.

Output

Output a single integer — the maximum possible value.

Sample test(s)

input
5 3 5 2 4 3 1
output
12

input
4 2 7 4 3 7
output
8

Note

Consider the first sample test. The optimal solution is obtained if the first subarray contains the first element only, the second subarray spans the next three elements and the last subarray contains the last element only. The sums of these subarrays are 5, 9 and 1, correspondingly.

Consider the second sample test. In the optimal solution, the first subarray consists of the first two elements and the second subarray consists of the third element only. Note that the last element does not belong to any subarray in this solution.

E2. Subarray Cuts

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

You are given an array of length n and a number k. Let's pick k non-overlapping non-empty subarrays of the initial array. Let s_i be the sum of the i-th subarray in order from left to right. Compute the maximum value of the following expression:

$$|s_1 - s_2| + |s_2 - s_3| + \ldots + |s_{k-1} - s_k|$$

Here subarray is a contiguous part of an array.

Input

The first line of input contains two integers n and k. The second line contains n integers — the elements of the array. The absolute values of elements do not exceed 10^4.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem E1 (9 points), constraints $2 \le n \le 400$, $2 \le k \le min(n, 50)$ will hold.
- In subproblem E2 (12 points), constraints $2 \le n \le 30000$, $2 \le k \le min(n, 200)$ will hold.

Output

Output a single integer — the maximum possible value.

Sample test(s)

input
5 3 5 2 4 3 1
output
12

input
4 2 7 4 3 7
output
8

Note

Consider the first sample test. The optimal solution is obtained if the first subarray contains the first element only, the second subarray spans the next three elements and the last subarray contains the last element only. The sums of these subarrays are 5, 9 and 1, correspondingly.

Consider the second sample test. In the optimal solution, the first subarray consists of the first two elements and the second subarray consists of the third element only. Note that the last element does not belong to any subarray in this solution.

F1. Scaygerboss

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

Cthulhu decided to catch Scaygerboss. Scaygerboss found it out and is trying to hide in a pack of his scaygers. Each scayger except Scaygerboss is either a male or a female. Scaygerboss's gender is "other".

Scaygers are scattered on a two-dimensional map divided into cells. A scayger looks nerdy and loveable if it is staying in the same cell with exactly one scayger of a gender that is different from its own gender. Cthulhu will not be able to catch Scaygerboss if all the scaygers on the map look nerdy and loveable.

The scaygers can move around at different speeds. For each scayger, we are given the time it takes this scayger to move from a cell to an adjacent cell. Cells are adjacent if they share a common side. At any point of time, each cell that does not contain an obstacle can be occupied by an arbitrary number of scaygers. Scaygers cannot move to cells with obstacles.

Calculate minimal time in order to make all scaygers look nerdy and loveable if they move optimally toward this goal.

Input

The first line contains 4 integers: n, m, $males$, $females$ ($0 \le males, females \le n \cdot m$). n and m are dimensions of the map; $males$ and $females$ are numbers of male scaygers and female scaygers.

Next n lines describe the map. Each of these lines contains m characters. Character '.' stands for a free cell; character '#' stands for a cell with an obstacle.

The next line contains 3 integers r, c, and t ($1 \le r \le n$, $1 \le c \le m$, $1 \le t \le 10^9$): the current coordinates of Scaygerboss and the time it takes Scaygerboss to move to an adjacent cell. The next $males$ lines contain coordinates and times of male scaygers in the same format as for Scaygerboss. The next $females$ lines contain coordinates and times of female scaygers in the same format as for Scaygerboss. (The coordinates and times adhere to the same limits as for Scaygerboss.) All scaygers reside in cells without obstacles.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem F1 (14 points), the constraints $1 \le n, m \le 11$ will hold.
- In subproblem F2 (6 points), the constraints $1 \le n, m \le 22$ will hold.

Output

Output the minimum possible time it takes to make all scaygers look nerdy and loveable or -1 if it is impossible.

Sample test(s)

input

```
4 4 2 3
....
.###
####
####
2 1 1
2 1 2
2 1 2
2 1 2
2 1 2
1 1 2
```

output

```
2
```

input

```
2 4 2 2
....
.###
2 1 1
2 1 2
2 1 2
2 1 2
2 1 2
```

output

```
-1
```

Note

Consider the first sample test. The scaygers are hiding on a 4 by 4 map. Scaygerboss initially resides in the cell $(2, 1)$ and can move between cells in 1 unit of time. There are also 2 male and 3 female scaygers on the map. One of the females initially is in the cell $(1, 1)$, and all the other scaygers are in the cell $(2, 1)$. All the scaygers move between cells in 2 units of time. If Scaygerboss and the female scayger from the cell $(1, 1)$ move to the cell $(1, 2)$, and a male and a female scayger from those residing in the cell $(2, 1)$ move to the cell $(1, 1)$, then all the scaygers will look nerdy and lovable in 2 units of time.

F2. Scaygerboss

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

Cthulhu decided to catch Scaygerboss. Scaygerboss found it out and is trying to hide in a pack of his scaygers. Each scayger except Scaygerboss is either a male or a female. Scaygerboss's gender is "other".

Scaygers are scattered on a two-dimensional map divided into cells. A scayger looks nerdy and loveable if it is staying in the same cell with exactly one scayger of a gender that is different from its own gender. Cthulhu will not be able to catch Scaygerboss if all the scaygers on the map look nerdy and loveable.

The scaygers can move around at different speeds. For each scayger, we are given the time it takes this scayger to move from a cell to an adjacent cell. Cells are adjacent if they share a common side. At any point of time, each cell that does not contain an obstacle can be occupied by an arbitrary number of scaygers. Scaygers cannot move to cells with obstacles.

Calculate minimal time in order to make all scaygers look nerdy and loveable if they move optimally toward this goal.

Input

The first line contains 4 integers: n, m, $males$, $females$ ($0 \le males, females \le n \cdot m$). n and m are dimensions of the map; $males$ and $females$ are numbers of male scaygers and female scaygers.

Next n lines describe the map. Each of these lines contains m characters. Character '.' stands for a free cell; character '#' stands for a cell with an obstacle.

The next line contains 3 integers r, c, and t ($1 \le r \le n$, $1 \le c \le m$, $1 \le t \le 10^9$): the current coordinates of Scaygerboss and the time it takes Scaygerboss to move to an adjacent cell. The next $males$ lines contain coordinates and times of male scaygers in the same format as for Scaygerboss. The next $females$ lines contain coordinates and times of female scaygers in the same format as for Scaygerboss. (The coordinates and times adhere to the same limits as for Scaygerboss.) All scaygers reside in cells without obstacles.

The problem consists of two subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem F1 (14 points), the constraints $1 \le n, m \le 11$ will hold.
- In subproblem F2 (6 points), the constraints $1 \le n, m \le 22$ will hold.

Output

Output the minimum possible time it takes to make all scaygers look nerdy and loveable or -1 if it is impossible.

Sample test(s)

input

```
4 4 2 3
....
.###
####
####
2 1 1
2 1 2
2 1 2
2 1 2
2 1 2
1 1 2
```

output

```
2
```

input

```
2 4 2 2
....
.###
2 1 1
2 1 2
2 1 2
2 1 2
2 1 2
```

output

```
-1
```

Note

Consider the first sample test. The scaygers are hiding on a 4 by 4 map. Scaygerboss initially resides in the cell $(2, 1)$ and can move between cells in 1 unit of time. There are also 2 male and 3 female scaygers on the map. One of the females initially is in the cell $(1, 1)$, and all the other scaygers are in the cell $(2, 1)$. All the scaygers move between cells in 2 units of time. If Scaygerboss and the female scayger from the cell $(1, 1)$ move to the cell $(1, 2)$, and a male and a female scayger from those residing in the cell $(2, 1)$ move to the cell $(1, 1)$, then all the scaygers will look nerdy and lovable in 2 units of time.

G1. Inversions problem

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

You are given a permutation of n numbers $p_1, p_2, ..., p_n$. We perform k operations of the following type: choose uniformly at random two indices l and r ($l \le r$) and reverse the order of the elements $p_l, p_{l+1}, ..., p_r$. Your task is to find the expected value of the number of inversions in the resulting permutation.

Input

The first line of input contains two integers n and k ($1 \le n \le 100$, $1 \le k \le 10^9$). The next line contains n integers $p_1, p_2, ..., p_n$ — the given permutation. All p_i are different and in range from 1 to n.

The problem consists of three subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem G1 (3 points), the constraints $1 \le n \le 6$, $1 \le k \le 4$ will hold.
- In subproblem G2 (5 points), the constraints $1 \le n \le 30$, $1 \le k \le 200$ will hold.
- In subproblem G3 (16 points), the constraints $1 \le n \le 100$, $1 \le k \le 10^9$ will hold.

Output

Output the answer with absolute or relative error no more than $1e-9$.

Sample test(s)

input
3 1 1 2 3
output
0.833333333333333

input
3 4 1 3 2
output
1.458333333333334

Note

Consider the first sample test. We will randomly pick an interval of the permutation $(1, 2, 3)$ (which has no inversions) and reverse the order of its elements. With probability $\frac{1}{2}$, the interval will consist of a single element and the permutation will not be altered. With probability $\frac{1}{6}$ we will inverse the first two elements' order and obtain the permutation $(2, 1, 3)$ which has one inversion. With the same probability we might pick the interval consisting of the last two elements which will lead to the permutation $(1, 3, 2)$ with one inversion. Finally, with probability $\frac{1}{6}$ the randomly picked interval will contain all elements, leading to the permutation $(3, 2, 1)$ with 3 inversions. Hence, the expected number of inversions is equal to $\frac{1}{2} \cdot 0 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 3 = \frac{5}{6}$.

G2. Inversions problem

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

You are given a permutation of n numbers $p_1, p_2, ..., p_n$. We perform k operations of the following type: choose uniformly at random two indices l and r ($l \le r$) and reverse the order of the elements $p_l, p_{l+1}, ..., p_r$. Your task is to find the expected value of the number of inversions in the resulting permutation.

Input

The first line of input contains two integers n and k ($1 \le n \le 100$, $1 \le k \le 10^9$). The next line contains n integers $p_1, p_2, ..., p_n$ — the given permutation. All p_i are different and in range from 1 to n.

The problem consists of three subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem G1 (3 points), the constraints $1 \le n \le 6$, $1 \le k \le 4$ will hold.
- In subproblem G2 (5 points), the constraints $1 \le n \le 30$, $1 \le k \le 200$ will hold.
- In subproblem G3 (16 points), the constraints $1 \le n \le 100$, $1 \le k \le 10^9$ will hold.

Output

Output the answer with absolute or relative error no more than $1e-9$.

Sample test(s)

input
3 1 1 2 3
output
0.833333333333333

input
3 4 1 3 2
output
1.458333333333334

Note

Consider the first sample test. We will randomly pick an interval of the permutation $(1, 2, 3)$ (which has no inversions) and reverse the order of its elements. With probability $\frac{1}{2}$, the interval will consist of a single element and the permutation will not be altered. With probability $\frac{1}{6}$ we will inverse the first two elements' order and obtain the permutation $(2, 1, 3)$ which has one inversion. With the same probability we might pick the interval consisting of the last two elements which will lead to the permutation $(1, 3, 2)$ with one inversion. Finally, with probability $\frac{1}{6}$ the randomly picked interval will contain all elements, leading to the permutation $(3, 2, 1)$ with 3 inversions. Hence, the expected number of inversions is equal to $\frac{1}{2} \cdot 0 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 3 = \frac{5}{6}$.

G3. Inversions problem

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

You are given a permutation of n numbers $p_1, p_2, ..., p_n$. We perform k operations of the following type: choose uniformly at random two indices l and r ($l \le r$) and reverse the order of the elements $p_l, p_{l+1}, ..., p_r$. Your task is to find the expected value of the number of inversions in the resulting permutation.

Input

The first line of input contains two integers n and k ($1 \le n \le 100$, $1 \le k \le 10^9$). The next line contains n integers $p_1, p_2, ..., p_n$ — the given permutation. All p_i are different and in range from 1 to n.

The problem consists of three subproblems. The subproblems have different constraints on the input. You will get some score for the correct submission of the subproblem. The description of the subproblems follows.

- In subproblem G1 (3 points), the constraints $1 \le n \le 6$, $1 \le k \le 4$ will hold.
- In subproblem G2 (5 points), the constraints $1 \le n \le 30$, $1 \le k \le 200$ will hold.
- In subproblem G3 (16 points), the constraints $1 \le n \le 100$, $1 \le k \le 10^9$ will hold.

Output

Output the answer with absolute or relative error no more than $1e-9$.

Sample test(s)

input
3 1 1 2 3

output
0.833333333333333

input
3 4 1 3 2

output
1.458333333333334

Note

Consider the first sample test. We will randomly pick an interval of the permutation $(1, 2, 3)$ (which has no inversions) and reverse the order of its elements. With probability $\frac{1}{2}$, the interval will consist of a single element and the permutation will not be altered. With probability $\frac{1}{6}$ we will inverse the first two elements' order and obtain the permutation $(2, 1, 3)$ which has one inversion. With the same probability we might pick the interval consisting of the last two elements which will lead to the permutation $(1, 3, 2)$ with one inversion. Finally, with probability $\frac{1}{6}$ the randomly picked interval will contain all elements, leading to the permutation $(3, 2, 1)$ with 3 inversions. Hence, the expected number of inversions is equal to $\frac{1}{2} \cdot 0 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 3 = \frac{5}{6}$.