Codeforces Round #182 (Div. 1)

A. Yaroslav and Sequence

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

Yaroslav has an array, consisting of $(2 \cdot n - 1)$ integers. In a single operation Yaroslav can change the sign of exactly n elements in the array. In other words, in one operation Yaroslav can select exactly n array elements, and multiply each of them by -1.

Yaroslav is now wondering: what maximum sum of array elements can be obtained if it is allowed to perform any number of described operations?

Help Yaroslav.

Input

The first line contains an integer n $(2 \leq n \leq 100)$. The second line contains $(2 \cdot n - 1)$ integers — the array elements. The array elements do not exceed 1000 in their absolute value.

Output

In a single line print the answer to the problem — the maximum sum that Yaroslav can get.

Sample test(s)

input
2 50 50 50

output
150

input
2 -1 -100 -1

output
100

Note

In the first sample you do not need to change anything. The sum of elements equals 150.

In the second sample you need to change the sign of the first two elements. Then we get the sum of the elements equal to 100.

B. Yaroslav and Time

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

Yaroslav is playing a game called "Time". The game has a timer showing the lifespan he's got left. As soon as the timer shows 0, Yaroslav's character dies and the game ends. Also, the game has n clock stations, station number i is at point (x_i, y_i) of the plane. As the player visits station number i, he increases the current time on his timer by a_i. The stations are for one-time use only, so if the player visits some station another time, the time on his timer won't grow.

A player spends $d \cdot dist$ time units to move between stations, where $dist$ is the distance the player has covered and d is some constant. The distance between stations i and j is determined as $|x_i - x_j| + |y_i - y_j|$.

Initially, the player is at station number 1, and the player has strictly more than zero and strictly less than one units of time. At station number 1 one unit of money can increase the time on the timer by one time unit (you can buy only integer number of time units).

Now Yaroslav is wondering, how much money he needs to get to station n. Help Yaroslav. Consider the time to buy and to increase the timer value negligibly small.

Input

The first line contains integers n and d $(3 \le n \le 100, 10^3 \le d \le 10^5)$ — the number of stations and the constant from the statement.

The second line contains $n - 2$ integers: $a_2, a_3, ..., a_{n-1}$ $(1 \le a_i \le 10^3)$. The next n lines contain the coordinates of the stations. The i-th of them contains two integers x_i, y_i $(-100 \le x_i, y_i \le 100)$.

It is guaranteed that no two stations are located at the same point.

Output

In a single line print an integer — the answer to the problem.

Sample test(s)

input
3 1000 1000 0 0 0 1 0 3
output
2000

input
3 1000 1000 1 0 1 1 1 2
output
1000

C. Yaroslav and Algorithm

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

Yaroslav likes algorithms. We'll describe one of his favorite algorithms.

1. The algorithm receives a string as the input. We denote this input string as a.
2. The algorithm consists of some number of command. Command number i looks either as $s_i >> w_i$, or as $s_i <> w_i$, where s_i and w_i are some possibly empty strings of length at most 7, consisting of digits and characters "?".
3. At each iteration, the algorithm looks for a command with the minimum index i, such that s_i occurs in a as a substring. If this command is not found the algorithm terminates.
4. Let's denote the number of the found command as k. In string a the first occurrence of the string s_k is replaced by string w_k. If the found command at that had form $s_k >> w_k$, then the algorithm continues its execution and proceeds to the next iteration. Otherwise, the algorithm terminates.
5. The value of string a after algorithm termination is considered to be the output of the algorithm.

Yaroslav has a set of n positive integers, he needs to come up with his favorite algorithm that will increase each of the given numbers by one. More formally, if we consider each number as a string representing the decimal representation of the number, then being run on each of these strings separately, the algorithm should receive the output string that is a recording of the corresponding number increased by one.

Help Yaroslav.

Input

The first line contains integer n $(1 \le n \le 100)$ — the number of elements in the set. The next n lines contains one positive integer each. All the given numbers are less than 10^{25}.

Output

Print the algorithm which can individually increase each number of the set. In the i-th line print the command number i without spaces.

Your algorithm will be launched for each of these numbers. The answer will be considered correct if:

- Each line will a correct algorithm command (see the description in the problem statement).
- The number of commands should not exceed 50.
- The algorithm will increase each of the given numbers by one.
- To get a respond, the algorithm will perform no more than 200 iterations for each number.

Sample test(s)

input
2 10 79
output
10<>11 79<>80

D. Yaroslav and Divisors

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

Yaroslav has an array $p = p_1, p_2, ..., p_n$ $(1 \le p_i \le n)$, consisting of n distinct integers. Also, he has m queries:

- Query number i is represented as a pair of integers l_i, r_i $(1 \le l_i \le r_i \le n)$.
- The answer to the query l_i, r_i is the number of pairs of integers q, w $(l_i \le q, w \le r_i)$ such that p_q is the divisor of p_w.

Help Yaroslav, answer all his queries.

Input

The first line contains the integers n and m $(1 \le n, m \le 2 \cdot 10^5)$. The second line contains n distinct integers $p_1, p_2, ..., p_n$ $(1 \le p_i \le n)$. The following m lines contain Yaroslav's queries. The i-th line contains integers l_i, r_i $(1 \le l_i \le r_i \le n)$.

Output

Print m integers — the answers to Yaroslav's queries in the order they appear in the input.

Please, do not use 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
1 1 1 1 1

output
1

input
10 9 1 2 3 4 5 6 7 8 9 10 1 10 2 9 3 8 4 7 5 6 2 2 9 10 5 10 4 10

output
27 14 8 4 2 1 2 7 9

E. Yaroslav and Arrangements

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

Yaroslav calls an array of r integers $a_1, a_2, ..., a_r$ *good*, if it meets the following conditions: $|a_1 - a_2| = 1, |a_2 - a_3| = 1, ..., |a_{r-1} - a_r| = 1, |a_r - a_1| = 1$, at that $a_1 = \min_{i=1}^{r} a_i$.

An array of integers $b_1, b_2, ..., b_r$ is called *great*, if it meets the following conditions:

1. The elements in it do not decrease $(b_i \le b_{i+1})$.
2. If the inequalities $1 \le r \le n$ and $1 \le b_i \le m$ hold.
3. If we can rearrange its elements and get at least one and at most k distinct good arrays.

Yaroslav has three integers n, m, k. He needs to count the number of distinct great arrays. Help Yaroslav! As the answer may be rather large, print the remainder after dividing it by $1000000007\ (10^9 + 7)$.

Two arrays are considered distinct if there is a position in which they have distinct numbers.

Input

The single line contains three integers $n, m, k\ (1 \le n, m, k \le 100)$.

Output

In a single line print the remainder after dividing the answer to the problem by number $1000000007\ (10^9 + 7)$.

Sample test(s)

input
1 1 1
output
0

input
3 3 3
output
2