Kotlin Heroes: Practice 2

A. Wrong Subtraction

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

Little girl Tanya is learning how to decrease a number by one, but she does it wrong with a number consisting of two or more digits. Tanya subtracts one from a number by the following algorithm:

- if the last digit of the number is non-zero, she decreases the number by one;
- if the last digit of the number is zero, she divides the number by 10 (i.e. removes the last digit).

You are given an integer number n. Tanya will subtract one from it k times. Your task is to print the result after all k subtractions.

It is guaranteed that the result will be positive integer number.

Input
The first line of the input contains two integer numbers n and k ($2 \le n \le 10^9$, $1 \le k \le 50$) — the number from which Tanya will subtract and the number of subtractions correspondingly.

Output
Print one integer number — the result of the decreasing n by one k times.

It is guaranteed that the result will be positive integer number.

Examples

input
512 4
output
50

input
1000000000 9
output
1

Note
The first example corresponds to the following sequence: $512 \to 511 \to 510 \to 51 \to 50$.

B. Two-gram

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

Two-gram is an ordered pair (i.e. string of length two) of capital Latin letters. For example, "AZ", "AA", "ZA" — three distinct two-grams.

You are given a string s consisting of n capital Latin letters. Your task is to find **any** two-gram contained in the given string **as a substring** (i.e. two consecutive characters of the string) maximal number of times. For example, for string s = "BBAABBBA" the answer is two-gram "BB", which contained in s three times. In other words, find any most frequent two-gram.

Note that occurrences of the two-gram can overlap with each other.

Input
The first line of the input contains integer number n ($2 \le n \le 100$) — the length of string s. The second line of the input contains the string s consisting of n capital Latin letters.

Output
Print the only line containing exactly two capital Latin letters — **any** two-gram contained in the given string s **as a substring** (i.e. two consecutive characters of the string) maximal number of times.

Examples

input

7 ABACABA
output
AB

input
5 ZZZAA
output
ZZ

Note
In the first example "BA" is also valid answer.

In the second example the only two-gram "ZZ" can be printed because it contained in the string "ZZZAA" two times.

C. Less or Equal

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

You are given a sequence of integers of length n and integer number k. You should print **any integer** number x in the range of $[1; 10^9]$ (i.e. $1 \le x \le 10^9$) such that exactly k elements of given sequence are less than or equal to x.

Note that the sequence can contain equal elements.

If there is no such x, print "-1" (without quotes).

Input
The first line of the input contains integer numbers n and k ($1 \le n \le 2 \cdot 10^5$, $0 \le k \le n$). The second line of the input contains n integer numbers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the sequence itself.

Output
Print **any integer** number x from range $[1; 10^9]$ such that exactly k elements of given sequence is less or equal to x.

If there is no such x, print "-1" (without quotes).

Examples

input
7 4 3 7 5 1 10 3 20
output
6

input
7 2 3 7 5 1 10 3 20
output
-1

Note
In the first example 5 is also a valid answer because the elements with indices $[1, 3, 4, 6]$ is less than or equal to 5 and obviously less than or equal to 6.

In the second example you cannot choose any number that only 2 elements of the given sequence will be less than or equal to this number because 3 elements of the given sequence will be also less than or equal to this number.

D. Divide by three, multiply by two

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

Polycarp likes to play with numbers. He takes some integer number x, writes it down on the board, and then performs with it $n - 1$ operations of the two kinds:

- divide the number x by 3 (x must be divisible by 3);
- multiply the number x by 2.

After each operation, Polycarp writes down the result on the board and replaces x by the result. So there will be n numbers on the board after all.

You are given a sequence of length n — the numbers that Polycarp wrote down. This sequence is given in arbitrary order, i.e. the order of the sequence can mismatch the order of the numbers written on the board.

Your problem is to rearrange (reorder) elements of this sequence in such a way that it can match possible Polycarp's game in the order of the numbers written on the board. I.e. each next number will be exactly two times of the previous number or exactly one third of previous number.

It is guaranteed that the answer exists.

Input
The first line of the input contatins an integer number n ($2 \le n \le 100$) — the number of the elements in the sequence. The second line of the input contains n integer numbers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 3 \cdot 10^{18}$) — rearranged (reordered) sequence that Polycarp can wrote down on the board.

Output
Print n integer numbers — rearranged (reordered) input sequence that can be the sequence that Polycarp could write down on the board.

It is guaranteed that the answer exists.

Examples

input
6 4 8 6 3 12 9
output
9 3 6 12 4 8

input
4 42 28 84 126
output
126 42 84 28

input
2 1000000000000000000 3000000000000000000
output
3000000000000000000 1000000000000000000

Note
In the first example the given sequence can be rearranged in the following way: $[9, 3, 6, 12, 4, 8]$. It can match possible Polycarp's game which started with $x = 9$.

E. Booking System

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

Innovation technologies are on a victorious march around the planet. They integrate into all spheres of human activity!

A restaurant called "Dijkstra's Place" has started thinking about optimizing the booking system.

There are n booking requests received by now. Each request is characterized by two numbers: c_i and p_i — the size of the group of visitors who will come via this request and the total sum of money they will spend in the restaurant, correspondingly.

We know that for each request, all c_i people want to sit at the same table and are going to spend the whole evening in the restaurant, from the opening moment at 18:00 to the closing moment.

Unfortunately, there only are k tables in the restaurant. For each table, we know r_i — the maximum number of people who can sit at it. A table can have only people from the same group sitting at it. If you cannot find a large enough table for the whole group, then all visitors leave and naturally, pay nothing.

Your task is: given the tables and the requests, decide which requests to accept and which requests to decline so that the money paid by the happy and full visitors was maximum.

Input
The first line of the input contains integer n ($1 \le n \le 1000$) — the number of requests from visitors. Then n lines follow. Each line contains two integers: c_i, p_i ($1 \le c_i, p_i \le 1000$) — the size of the group of visitors who will come by the i-th request and the total

sum of money they will pay when they visit the restaurant, correspondingly.

The next line contains integer k ($1 \le k \le 1000$) — the number of tables in the restaurant. The last line contains k space-separated integers: $r_1, r_2, \ldots, r_k$ ($1 \le r_i \le 1000$) — the maximum number of people that can sit at each table.

Output

In the first line print two integers: m, s — the number of accepted requests and the total money you get from these requests, correspondingly.

Then print m lines — each line must contain two space-separated integers: the number of the accepted request and the number of the table to seat people who come via this request. The requests and the tables are consecutively numbered starting from 1 in the order in which they are given in the input.

If there are multiple optimal answers, print any of them.

Examples

input
3 10 50 2 100 5 30 3 4 6 9
output
2 130 2 1 3 2

F. One-Based Arithmetic

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

Prof. Vasechkin wants to represent positive integer n as a sum of addends, where each addends is an integer number containing only 1s. For example, he can represent 121 as 121=111+11+–1. Help him to find the least number of digits 1 in such sum.

Input

The first line of the input contains integer n ($1 \le n < 10^{15}$).

Output

Print expected minimal number of digits 1.

Examples

input
121
output
6

G. Hiking

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

A traveler is planning a water hike along the river. He noted the suitable rest points for the night and wrote out their distances from the starting point. Each of these locations is further characterized by its *picturesqueness*, so for the i-th rest point the distance from the start equals x_i, and its *picturesqueness* equals b_i. The traveler will move down the river in one direction, we can assume that he will start from point 0 on the coordinate axis and rest points are points with coordinates x_i.

Every day the traveler wants to cover the distance l. In practice, it turns out that this is not always possible, because he needs to end each day at one of the resting points. In addition, the traveler is choosing between two desires: cover distance l every day and visit the most picturesque places.

Let's assume that if the traveler covers distance r_j in a day, then he feels frustration $\sqrt{|r_j - l|}$, and his total frustration over the hike is calculated as the total frustration on all days.

Help him plan the route so as to minimize the *relative total frustration*: the total frustration divided by the total picturesqueness of all the rest points he used.

The traveler's path must end in the farthest rest point.

Input
The first line of the input contains integers n, l ($1 \le n \le 1000$, $1 \le l \le 10^5$) — the number of rest points and the optimal length of one day path.

Then n lines follow, each line describes one rest point as a pair of integers x_i, b_i ($1 \le x_i, b_i \le 10^6$). No two rest points have the same x_i, the lines are given in the order of strictly increasing x_i.

Output
Print the traveler's path as a sequence of the numbers of the resting points he used in the order he used them. Number the points from 1 to n in the order of increasing x_i. The last printed number must be equal to n.

Examples

input
5 9 10 10 20 10 30 1 31 5 40 10
output
1 2 4 5

Note
In the sample test the minimum value of *relative total frustration* approximately equals 0.097549. This value can be calculated as $(1 + 1 + \sqrt{2} + 0)/(10 + 10 + 5 + 10)$.

H. Berland Federalization

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

Recently, Berland faces federalization requests more and more often. The proponents propose to divide the country into separate states. Moreover, they demand that there is a state which includes exactly k towns.

Currently, Berland has n towns, some pairs of them are connected by bilateral roads. Berland has only $n - 1$ roads. You can reach any city from the capital, that is, the road network forms a *tree*.

The Ministry of Roads fears that after the reform those roads that will connect the towns of different states will bring a lot of trouble.

Your task is to come up with a plan to divide the country into states such that:

- each state is connected, i.e. for each state it is possible to get from any town to any other using its roads (that is, the roads that connect the state towns),
- there is a state that consisted of exactly k cities,
- the number of roads that connect different states is minimum.

Input
The first line contains integers n, k ($1 \le k \le n \le 400$). Then follow $n - 1$ lines, each of them describes a road in Berland. The roads are given as pairs of integers x_i, y_i ($1 \le x_i, y_i \le n$; $x_i \ne y_i$) — the numbers of towns connected by the road. Assume that the towns are numbered from 1 to n.

Output
The the first line print the required minimum number of "problem" roads t. Then print a sequence of t integers — their indices in the found division. The roads are numbered starting from 1 in the order they follow in the input. If there are multiple possible solutions, print any of them.

If the solution shows that there are no "problem" roads at all, print a single integer 0 and either leave the second line empty or do not print it at all.

Examples

input
5 2 1 2 2 3 3 4 4 5
output
1 2

input
5 3

1 2
1 3
1 4
1 5

output

2
3 4

input

1 1

output

0