Problem A. SDU Open

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In a parallel universe, *SDU Open* is a new international olympiad for school students in the field of computer science. It aims to attract talented students from different countries of the world and give them the opportunity to demonstrate their knowledge and skills in competition.

According to the traditions of international olympiads, in the first year of *SDU Open*, medals will be distributed as follows:

- At least 1/12 of the participants should receive gold medals;
- At least 1/4 of the participants should receive silver medals or higher;
- At least 1/2 of the participants should receive bronze medals or higher.

If it is possible to distribute medals in several ways, the option with the smallest number of gold medals is chosen. If there are several such options, the option with the smallest number of silver medals is chosen. If there are several such options, the option with the smallest number of bronze medals is chosen.

Find the number of gold, silver, and bronze medals that should be distributed if n participants have registered for the olympiad.

Input

The first line contains a single integer n ($1 \leq n \leq 1000$) - the number of participants in the olympiad.

Output

Output three integers separated by a space - the number of gold, silver, and bronze medals that will be distributed at the olympiad.

Examples

standard input	standard output
1	1 0 0
12	1 2 3
1000	84 166 250

Problem B. From decreasing to increasing

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A permutation of length n is a sequence of all integers from 1 to n inclusive, arranged in a certain order. In other words, it is a one-to-one mapping of numbers from 1 to n onto themselves.

You have a permutation p of length n, sorted in *decreasing* order. You want to sort this permutation in *increasing* order. To do this, you can perform the following operation:

- Choose two consecutive blocks of the same size and swap them. In other words, you choose two integers (s, k) such that $1 \le s, k \le n$ and $s + 2k - 1 \le n$, and swap the blocks $(p_s, \ldots, p_{s+k-1})$ and $(p_{s+k}, \ldots, p_{s+2k-1})$.

For example, if $p = (2, 3, 1, 4, 5, 7, 6)$ and you perform the operation with the pair $s = 2$ and $k = 2$, you will get the permutation $p = (2, 4, 5, 3, 1, 7, 6)$ by swapping the consecutive blocks $(3, 1)$ and $(4, 5)$ of length 2.

Sort p in increasing order using no more than n operations.

Input

The first line of the input file contains a single integer n $(1 \le n \le 1000)$.

Output

In the first line, output a single integer m $(0 \le m \le n)$ — the number of operations you performed.

In the next m lines, output pairs (s_i, k_i) — all the operations you performed in the order they were performed.

Examples

standard input	standard output

`6`	`4`
	`1 3`
	`1 2`
	`2 2`
	`3 2`

`2`	`1`
	`1 1`

Note

Explanation for the first example:

1. $(\mathbf{6,\ 5,\ 4,\ 3,\ 2,\ 1}) \to (3, 2, 1, 6, 5, 4)$
2. $(\mathbf{3,\ 2,\ 1,\ 6}, 5, 4) \to (1, 6, 3, 2, 5, 4)$
3. $(1, \mathbf{6,\ 3,\ 2,\ 5}, 4) \to (1, 2, 5, 6, 3, 4)$
4. $(1, 2, \mathbf{5,\ 6,\ 3,\ 4}) \to (1, 2, 3, 4, 5, 6)$

Problem C. Serial Representative

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Let us define a *series* of a binary string as the maximum continuous substring of this string consisting of only one character, either 0 or 1. For example, in the binary string '`110100011001`' there are seven series: '`11`', '`0`', '`1`', '`000`', '`11`', '`00`' and '`1`'.

Take an arbitrary binary string S of length n and consider all its series. Sort all its series consisting of zeros in non-decreasing order of length. Similarly, sort all series consisting of ones. Then combine all these series preserving the relative order between blocks of zeros and ones and obtain a new string S'. We call this new string the *serial representative* of the string S.

For example, let's do this procedure for the string '`110100011001`' given above:

- All series consisting of zeros: '`0`', '`000`', '`00`'. Sort by length and get '`0`', '`00`', '`000`'.
- All series consisting of ones: '`11`', '`1`', '`11`', '`1`'. Sort by length and get '`1`', '`1`', '`11`', '`11`'.
- Combine the obtained series preserving the relative order between blocks of zeros and ones. We get '`1`', '`0`', '`1`', '`00`', '`11`', '`000`', '`11`'. Write it as a string and get '`101001100011`'. This string is the *serial representative* of the original string '`110100011001`'.

You are given one integer n. Your task is to count the number of binary strings S of length n such that its *serial representative* S' is lexicographically smaller than the string S ($S' < S$). Since the answer can be very large, calculate it modulo 998 244 353.

For two strings a and b of length n, a is lexicographically smaller than b if and only if there exists a position i such that $a_i < b_i$ and $a_j = b_j$ for all $j < i$.

Input

The first line of the input file contains a single integer n ($1 \le n \le 5000$).

Output

Output a single integer — the number of required strings modulo 998 244 353.

Examples

standard input	standard output
3	0
4	1

Note

The only binary string of length 4 for which $S' < S$ holds is '`1101`'. Its serial representative is the string '`1011`'.

Problem D. UFC

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Tima and Batyr are playing a series of matches in the game UFC. The game has n fighters, numbered in order of their strength from 1 (weakest) to n (strongest).

It is planned to hold n matches in such a way that each player plays exactly once with each fighter. Before the start of the series, Tima and Batyr choose the order in which they will play with their fighters.

In each match, the winner is determined by the strength of the fighter chosen by the player. If both players choose the same fighter, the match is considered a draw, and Tima is declared the winner due to his age.

If at any point in the series Batyr wins more matches than Tima, he declares himself the stronger player and goes home. However, if all n matches are played and Batyr has not gone home, he acknowledges Tima's strength.

As the most loyal fan of Tima, you became interested: how many ways are there to choose the order of fighters for the players so that Tima beats Batyr? Since the answer can be very large, calculate it modulo 998 244 353.

Input

The first line contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the number of fighters and, correspondingly, matches.

Output

Print a single number — the number of ways modulo 998 244 353.

Examples

standard input	standard output
1	1
2	3
3	22
4	310

Note

In the second example, there are only 4 possible ways to choose the order of fighters. The only one where Batyr wins is the one where Batyr chooses the order of fighters $(2, 1)$ and Tima chooses the order of fighters $(1, 2)$. In this case, Batyr wins the first match, immediately declares himself the winner and goes home.

In the fourth example, there are 576 possible ways to choose the order of fighters. If Batyr chooses the order of fighters $(1, 4, 3, 2)$ and Tima chooses the order of fighters $(1, 3, 2, 4)$, then the series will proceed as follows:

1. Batyr and Tima both chose fighter 1, so Tima wins due to his age;
2. Batyr chose fighter 4 and Tima chose fighter 3, so Batyr wins;
3. Batyr chose fighter 3 and Tima chose fighter 2, so Batyr wins. At this point, Batyr has already won more matches than Tima (2 to 1), so Batyr declares himself the winner and goes home.

Problem E. Cultural Dissonance

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Batyr and Mazen decided to go to a popular place where they cook wok and offer n different toppings. Each topping changes the taste of the wok by a_i, which can also be a negative number. Wok without any toppings has a taste of 0.

They want to try all possible woks, which is a total of 2^n. Batyr wants to write down a list of the order in which they will try all the woks. Then he will share this list with Mazen. However, since Mazen's native language is Arabic, he will read this list from right to left. This means that the first wok in Batyr's list will be the last in Mazen's list and vice versa.

To avoid disagreements or conflicts, Batyr wants to create the list in such a way that the woks they will try at the same time always have the same taste for both him and Mazen. In other words, Batyr needs to ensure that the tastes of the woks are identical when reading the list from left to right and from right to left.

Determine if Batyr can create such a list.

Input

The first line contains a single integer n ($1 \le n \le 30$) - the number of toppings.

The second line contains n integers $a_1, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$) - the tastes of the toppings.

Output

If it is possible to create a list, print '`YES`'. Otherwise, print '`NO`'.

Examples

standard input	standard output
2 1 -2	NO
3 -1 0 1	YES

Note

In the first example, there are only 4 possible woks:

- Wok without toppings with a taste of 0.
- Wok with the first topping with a taste of 1.
- Wok with the second topping with a taste of -2.
- Wok with both toppings and a taste of -1.

It can be shown that there is no order in which the tastes of the woks would be identical when reading the list from left to right and from right to left.

Problem F. Lost in the Jungle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

It is very easy to get lost in dense jungles, and sometimes the only way to find a way out is to wander. Fortunately, very experienced travelers have visited our jungle and have laid out paths.

There are only n trees and $n-1$ paths, each of which connects two trees. Additionally, it is known that between any two trees there is exactly one route along the paths, without passing through the same path twice.

Over the past few days, q people have independently found themselves in the jungle without a map or other means of navigation. You know from the rescuers' records that the i-th person started his journey at tree s_i and wandered until he reached tree t_i for the first time, where he found a note from the rescuers with a way out of the jungle.

Each of these people wandered completely randomly, looking at all possible paths he could take at each step, and choosing any of them with equal probability, reaching the other end of the path and repeating this process. The probability of moving from tree u to the tree connected by a path v was $\frac{1}{c_u}$, where c_u is the total number of paths at tree u. Each of them never gave up and never stayed at the same tree, constantly walking along the paths and looking for a way out.

However, the records did not indicate how long each of these lost people wandered. To estimate this number, you need to find the expected number of steps required for each person to reach tree t_i from tree s_i.

Input

The first line contains two integers n and q ($2 \le n, q \le 2 \cdot 10^5$) — the number of trees in the jungle and the number of lost people.

The next $n-1$ lines contain descriptions of the paths. Each line contains two integers u and v ($1 \le u, v \le n$), meaning that trees u and v are connected by a path.

The next q lines contain two integers s_i and t_i ($1 \le s_i, t_i \le n$, $s_i \ne t_i$) — the initial and final tree for each lost person.

Output

Output q lines, each containing one integer — the expected number of steps modulo 998 244 353 required for each person to reach tree t_i from tree s_i.

Formally, let $M = 998\,244\,353$. It can be shown that the answer can be represented as an irreducible fraction $\frac{p}{q}$, where p and q are integers. Output one integer, equal to $(p \cdot q^{-1}) \bmod M$.

Example

standard input	standard output

3 3
1 2
1 3
2 3
1 2
2 1
	4
	3
	1

Problem G. GCD Encryption

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Greatest Common Divisor (GCD) is a fundamental concept in mathematics that helps us understand the relationship between two numbers. It is the largest positive integer that divides two different numbers without leaving a remainder.

Finding GCD is an important skill that can be used in many areas of mathematics, including algebra, number theory, and cryptography. For example, in cryptography, the security of some encryption schemes depends on the complexity of finding the GCD of two large prime numbers.

One day, Alex decided to challenge himself by inventing his own encryption algorithm using GCD. He studied various encryption methods and was inspired to create something new and unique.

Instead of using large prime numbers, he decided to encode his data with a single array of positive integers $a_1, \ldots, a_n$. This array only has meaning if it is ordered in a certain way, so Alex properly shuffled his array.

Alex knew that the original order had a very special property: the numbers $(a_i + i)$ **were not** coprime. More formally, the condition $\mathrm{GCD}(a_1 + 1, a_2 + 2, \ldots, a_n + n) \neq 1$ was satisfied.

Soon Alex realized that deciphering his cipher was far from an easy task. He already has a shuffled array $a_1, \ldots, a_n$ of length n. Help him restore the original order of the array or tell Alex that such an order does not exist and he made a mistake somewhere. If there can be multiple orders, any of them can be output.

Input

The first line of the input file contains a single integer n — the size of the array a $(2 \leq n \leq 10^5)$.

The second line of the input file contains n integers $a_1, \ldots, a_n$ $(1 \leq a_i \leq 10^9)$.

Output

If the given array cannot be decrypted, output `'NO'`. Otherwise, output `'YES'` on the first line and output the decrypted array $a'_1, \ldots, a'_n$ on the second line.

Examples

standard input	standard output
3 1 2 5	YES 5 2 1
3 2 2 3	NO

Note

$\mathrm{GCD}(5 + 1, 2 + 2, 1 + 3) = \mathrm{GCD}(6, 4, 4) = 2 \neq 1$. There is also a solution $(1, 2, 5)$ since $\mathrm{GCD}(1 + 1, 2 + 2, 5 + 3) = 2 \neq 1$.

Problem H. SDUcell

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

For mobile operators, it is extremely important to place radio towers as close to users as possible, as the distance between the tower and the user directly affects the signal strength. The further the user is from the tower, the weaker the signal will be, which can lead to call drops, low data transfer speeds, and overall low mobile network performance. By placing radio towers closer to users, mobile operators can guarantee that users will receive a strong and stable signal, which is necessary for providing high-quality mobile services and maintaining customer satisfaction.

A very well-known mobile operator, *SDUcell*, has n radio towers in the city. In this problem, the city can be represented as a two-dimensional Euclidean plane and the i-th radio tower is located at the integer coordinate (x_i, y_i). The distance between two points is measured by the Euclidean distance, which is the length of the shortest line segment connecting the given points.

An active user of this operator, Marco, decided to take a walk through the streets of the city and chat with friends during the walk. He has already determined his route and highlighted m points on the map.

He starts his route at point (a_1, b_1). Then he will walk in a straight line to point (a_2, b_2), from there to point (a_3, b_3) and so on. In the end, he will finish his route at point (a_m, b_m).

The peculiarity of his route is that the streets in the city are arranged in a rectangular shape. Since Marco always walks along the streets, each part of his route will be either strictly horizontal or strictly vertical. More formally, for all $1 \le i \le m - 1$, either $a_i = a_{i+1}$ or $b_i = b_{i+1}$ will always hold.

Marco moves at a constant speed of one unit per second. That is, the distance traveled is directly proportional to the time spent, and Marco will cover d units of distance in d seconds.

At each moment in time, the mobile operator *SDUcell* will connect Marco's phone to the nearest radio tower. At the end of the route, *SDUcell* calculates the cost of the call in a very interesting way:

- Let's denote the total travel time of Marco as d. We will assume that Marco started the call at time 0 and ended it at time d. For each moment in time t $(0 \le t \le d)$, we define the value $f(t)$ as the distance from Marco to the nearest radio tower at that moment.
- Then, the cost of the call in tenge will be determined as the area under the function $C(t) = f(t)^2$ in the interval $[0, d]$.

How much will Marco have to pay?

Input

The first line of the input file contains a single integer n $(1 \le n \le 2000)$ — the number of radio towers of the mobile operator *SDUcell*.

The next n lines contain the integer coordinates of the radio towers (x_i, y_i) $(-1000 \le x_i, y_i \le 1000)$. It is guaranteed that all coordinates are distinct.

The next line contains a single integer m $(1 \le m \le 500)$ — the number of points in Marco's route.

The next m lines contain the integer coordinates of all the points in Marco's route (a_i, b_i) $(-1000 \le a_i, b_i \le 1000)$. It is guaranteed that all points on the route are distinct. It is also guaranteed that for all $1 \le i \le m - 1$, either $a_i = a_{i+1}$ or $b_i = b_{i+1}$ holds.

Output

Output a single real number — the sum that Marco will have to pay. Your answer will be considered correct if its relative or absolute error does not exceed 10^{-6}.

Examples

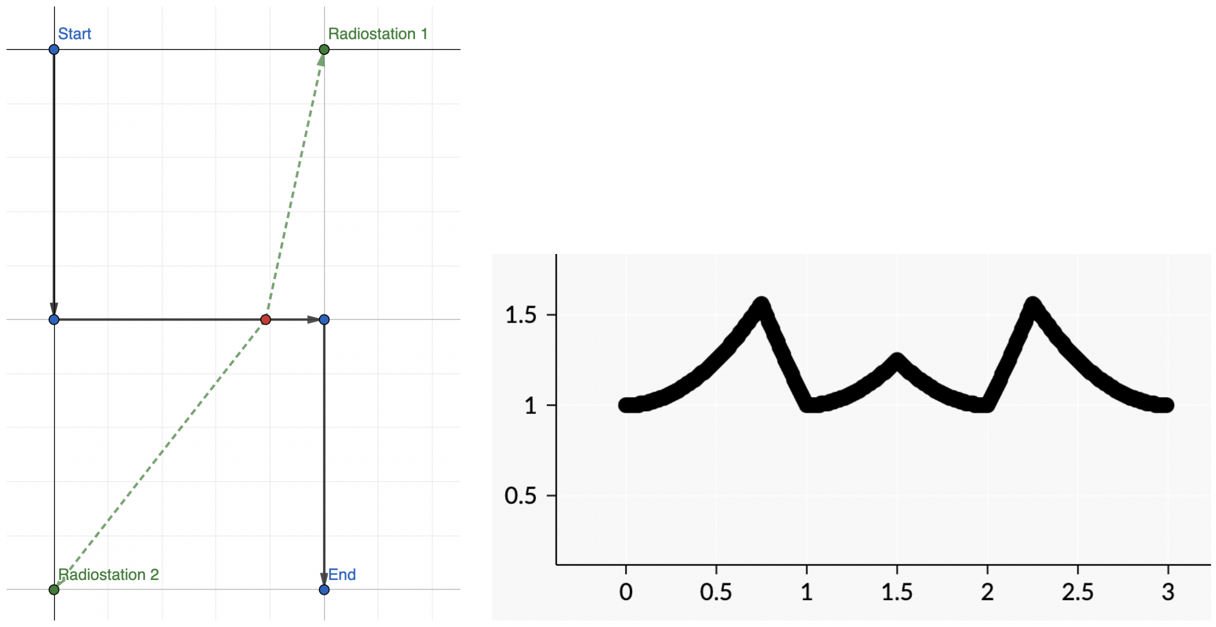
standard input	standard output

2
1 0
0 -2
4
0 0
0 -1
1 -1
1 -2
	3.500000000

3
0 0
2 2
0 9
2
0 0
0 9
	44.678571429

Note

Here is a visualization of the first example.



Marco's route, locations of radio towers, and the function $C(t)$

Problem I. Broken sword

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Daniyar is a brave knight who fights against n monsters arranged in a row. The i-th monster from the left has a strength of a_i.

Daniyar has a sword of length k, which allows him to kill no more than k consecutive monsters with one swing. After he kills the monsters, the remaining monsters move forward, closing the row again.

However, Daniyar can swing his sword no more than m times before getting tired and unable to continue the fight.

Daniyar's main goal is to defeat the strongest monsters before he gets too tired to continue the battle. Therefore, he wants to minimize the strength of the strongest monster that will still be alive after this battle.

Unfortunately, just before the battle, Daniyar broke his sword. To solve this problem, he urgently needs to go to the nearest blacksmith to buy a new sword. However, he is not sure about the length of the sword he should buy. He wants to determine the minimum strength of the strongest monster that would still be alive if he had a sword of length k for each possible value of k ($1 \le k \le n$).

Input

The first line contains two integers n and m ($1 \le n, m \le 3 \cdot 10^5$) — the number of monsters and the number of times Daniyar can swing his sword, respectively.

The second line contains n integers $a_1, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the strength of each monster.

Output

For each k ($1 \le k \le n$), output the minimum possible strength of the strongest monster that will still be alive after the battle if Daniyar had a sword of length k, or 0 if he can kill all the monsters.

Examples

standard input	standard output

`5 1`
`3 5 4 1 2`
	`4 3 2 2 0`

`3 2`
`3 1 2`
	`1 0 0`

Problem J. Chef Coder

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Egor is famous for his talent in sports programming, but few know that he is also an excellent cook who can easily prepare a variety of dishes.

He can be called CodeChef, given his impressive skills in solving FFT problems as well as in cooking steaks.

When the organizers of the SDU Open needed a chef to prepare steaks for the contest participants, they knew that Egor was the perfect person for the job.

Currently, there are n participants in the queue, where the i-th participant must eat at least a_i steaks to satisfy their hunger and eat b_i steaks to feel full.

Although Egor is determined to make everyone full, the organizers want to save on food. To find a compromise, Egor was tasked with serving steaks in such a way that all participants are satisfied and at any given time, the number of full participants is not greater than the number of satisfied ones.

If it is known that Egor can prepare no more than k steaks, what is the maximum number of participants he can make full?

Input

The first line contains two integers n and k ($1 \le n \le 10^5$, $1 \le k \le 10^9$) — the number of people in the queue and the maximum number of steaks Egor can prepare.

The next n lines contain two integers a_i and b_i ($1 \le a_i < b_i \le 10^9$) — the number of steaks for the i-th participant to satisfy their hunger and feel full, respectively.

Output

If it is impossible to satisfy the hunger of all participants, output `-1`. Otherwise, output a single integer — the maximum number of full participants.

Examples

standard input	standard output

3 2
1 2
1 3
1 4
	-1

3 4
1 2
1 3
1 4
	0

3 5
1 2
1 3
1 4
	1

Note

In the first example, Egor does not have enough steaks to satisfy the hunger of all participants ($2 < 3$).

In the second example, Egor can satisfy the first participant, but after that, the number of full participants will be greater than the number of satisfied ones, which goes against the compromise with the organizers. Therefore, Egor will have to not make the first participant full.

Problem K. Postal code

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

There are n different postal codes $a_1, \ldots, a_n$ in Barcelona. Each postal code is a 5-digit integer (possibly with leading zeros).

Exactly one post office is responsible for each postal code. All deliveries to a specific postal code always go to the office.

Some post offices know the contact numbers and locations of each other. To make it easier to remember, they agreed on the following:

- Post offices with numbers i and j ($1 \le i < j \le n$) will exchange contact numbers only if their postal codes a_i and a_j differ in exactly k digits.

Recently, Tommaso came to Barcelona on vacation to work as a courier. It is very important for couriers to know the contact numbers and locations of as many post offices as possible.

Since Tommaso is not familiar with the city, he decided to start with his place of residence. He will come to the post office of his place of residence and start asking for contacts and details of other post offices. Then he will visit those offices and ask them about even more offices and so on.

Tommaso does not yet know the postal code of his place of residence. You became interested: for all possible options s ($1 \le s \le n$) of the post office of Tommaso's place of residence, how many different post offices will Tommaso be able to find out about if he starts from there?

Input

The first line contains two integers n and k ($1 \le n \le 10^5$, $1 \le k \le 5$).

Each of the next n lines contains integers $a_1, \ldots, a_n$ ($00\,000 \le a_i \le 99\,999$) — all the different postal codes of Barcelona, written with exactly 5 digits.

Output

Output exactly n integers: the number of different post offices that Tommaso will be able to find out about if he starts from the office with number s ($1 \le s \le n$).

Examples

standard input	standard output

`4 1`
`02345`
`02355`
`12345`
`85475`
	`3 3 3 1`

`3 5`
`12345`
`25876`
`57805`
	`2 2 1`

Problem L. Dream Team

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There is a group of n undergraduate students at *SDU* who are interested in participating in the upcoming *ICPC* competitions. For each student, their strength level is known, denoted by a single integer a_i.

There are also m similar graduate students at the university who are interested in the role of coach. For each of them, their strength level is also known, denoted by a single integer b_i.

The *SDU* administration is interested in how potentially balanced a team can be assembled from their university. To do this, a team of three undergraduate students i, j, and q $(1 \le i < j < q \le n)$ must be formed. The strength of the team will be determined by the sum of the individual members of the team, namely — the number $a_i + a_j + a_q$.

However, the team will not develop fully without a coach. Therefore, it was decided to choose one coach from the graduate students. Let the number of the selected coach be k $(1 \le k \le m)$. Then the following conditions must also be met:

1. The coach must be stronger than the team so that he can train them. Therefore, the condition $b_k > a_i + a_j + a_q$ must be met.
2. The coach should not be too strong compared to the team so that the coach is not bored teaching, and the participants do not find it too difficult to train.

Taking into account the above restrictions, it was decided to form a suitable team so that the difference between the strength of the coach and the strength of the team $b_k - (a_i + a_j + a_q)$ was minimal. If it is possible to form several suitable teams, any of them can be chosen.

Help assemble the dream team.

Input

The first line contains two integers n and m $(1 \le n, m \le 300)$ — the number of undergraduate and graduate students, respectively.

The second line contains n integers $a_1, \ldots, a_n$ $(1 \le a_i \le 10000)$ — the strength levels of undergraduate students.

The third line contains m integers $b_1, \ldots, b_m$ $(1 \le b_i \le 10000)$ — the strength levels of graduate students.

Output

If it is possible to form the dream team, output four integers i, j, q, k $(1 \le i < j < q \le n$, $1 \le k \le m)$ — the numbers of the team members and the coach.

If it is impossible to assemble the dream team, output a single integer `-1`.

Examples

standard input	standard output
3 2	1 2 3 2
1 2 3	
10 8	
3 2	-1
1 2 3	
6 4	
5 5	2 3 5 3
1 2 4 8 16	
6 18 23 27 30	

Note

In the first example, only three undergraduate students can be selected for the team, and the strength of this team will be $1+2+3=6$. For the role of coach, we can choose a graduate student with a strength of 8 or a graduate student with a strength of 10, among them it is best to choose a student with a strength of 8 to minimize the difference between the strength of the team and the strength of the coach.

In the second example, the strength of the only possible team is 6, and it is impossible to choose a coach who would be stronger than this team.