Problem A. Job Applications

Time limit: 1 second
Memory limit: 256 megabytes

After finishing university, Anton decided to start looking for a job.

He immediately went to the `ConnectedIn` website. When he went to the "Vacancies" tab, he saw a lot of job openings. However, the exact number of vacancies was not specified on the website. He knows that there are a maximum of 20 vacancies on each page, and the vacancies take up n pages.

You have a good understanding of how the website works. When a new vacancy is added, it is added to the last page. If there are already 20 vacancies on the last page before adding a new one, a new page is created and the vacancy is added there.

Knowing this information, determine the minimum and maximum number of vacancies.

Input

The first line of the input contains a single integer n ($1 \le n \le 100$) — the number of pages with vacancies.

Output

Print two integers — the minimum and maximum number of vacancies that could be on the website.

Examples

standard input	standard output
3	41 60
14	261 280

Problem B. Tic-Tac-Toe

Time limit:	1 second
Memory limit:	256 megabytes

You were playing tic-tac-toe with a friend online. However, there was a problem, another air raid in your area! Being a responsible person, you go to the bomb shelter. After the alarm is over, you come back to finish the game, but something is not right. Your friend may have cheated and changed the board.

You remember the game board A as it was. Upon returning, you see the same game and the board B. Can you tell if it is possible to obtain board B from board A in no more than one move, made according to the rules?

Please note that the first move is made by the player placing the **X**. Also, the standard rule that the game stops if there are three X's or O's in a row does not apply here.

Input

The first three lines contain three symbols $A_{i,j}$ each, describing the initial board.

The next three lines contain three symbols $B_{i,j}$ each, describing the final board.

Each cell of the board is described by three symbols:

- "`.`" — denotes an empty cell;
- "`O`" — denotes a placed O;
- "`X`" — denotes a placed X.

It is guaranteed that board A can be obtained by a sequence of valid moves from an empty board.

Output

Output "`YES`" or "`NO`" (in any case) depending on whether it was possible to obtain board B from board A.

Examples

standard input	standard output
.X. .X. 00. .X. .X. 000	NO
.XX .00 ... XXX .00 ...	YES
XXX 000 ... XXX 000 .X.	YES
0X0 X.X 0X0 X0X 0.0 X0X	NO
.X. ... ... .X.	YES

Note

In the first example, one O is added, but it is now the turn to place an X.

In the second example, one X is added.

In the third example, one X is also added. Note that despite having three O's (and X's) in a row, the game does not stop.

In the fourth example, the board is altered.

In the fifth example, no moves were made.

Problem C. Disco

Time limit: 3 seconds
Memory limit: 256 megabytes

Anton, as a teacher at school, is organizing a disco for the children. He has been tasked with selecting a playlist so that everyone is thrilled with the celebration. But Anton has a huge database of songs, and unfortunately, the time for the disco is limited :(.

Specifically, there are n songs in the database. Each song can be characterized by two integers t_i and r_i — the length of the song and its rating. As a music lover, Anton wants to select **exactly** k songs (i.e., not fewer) to maximize the ratio of the sum of ratings to the sum of lengths.

More formally, let S be the set of songs and $X \subseteq S$, $|X| = k$ — a subset of songs that have been selected for the playlist. The goal is to maximize $\frac{\sum_{i \in X} r_i}{\sum_{i \in X} t_i}$. In other words, it is necessary to find the sum of the numbers r_i of all the songs that will be played at the disco, find the sum of the numbers t_i of all the songs that will be played at the disco, and divide the former by the latter. This number needs to be maximized.

Find and output the maximum possible ratio.

Input

The first line contains two integers n, k ($1 \le k \le n \le 10^5$) — the total number of songs and the number of songs to be selected for the playlist.

The second line contains n integers r_i ($1 \le r_i \le 10^5$).

The third line contains n integers t_i ($1 \le t_i \le 10^5$).

Output

Output a single number — the maximum ratio.

Your answer will be considered correct if its absolute or relative error does not exceed 10^{-4}.

Formally, let your answer be a, and the jury's answer be b. Your answer will be accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-4}$.

Examples

standard input	standard output
2 2 1 2 2 3	0.600000
5 2 1 2 3 2 3 2 4 2 5 3	1.200000

Note

In the first example, there are two songs that need to be added to the playlist. Therefore, the ratio will be:

$$\frac{1+2}{2+3} = \frac{3}{5} = 0.6$$

In the second example, it is best to take the third and fifth songs. Therefore, the ratio will be:

$$\frac{3+3}{2+3} = \frac{6}{5} = 1.2$$

Problem D. Sequence

Time limit: 1.5 seconds
Memory limit: 512 megabytes

Anton is under pressure — he has to submit all the assignments. As often happens — he cannot extend the deadline...

You are given a sequence a of n integers, and two integers l and r. You need to find the longest subsequence b of the sequence a such that $l \le b_i + b_{i+1} \le r$ $(1 \le i < |b|)$. Here, $|b|$ denotes the number of elements in the sequence b. In other words, you need to select a subsequence such that the sum of any two adjacent numbers is not less than l and not greater than r.

A subsequence of an array is a sequence that can be obtained by deleting several (possibly none) elements from the original sequence.

Input

The first line contains three integers n, l, r $(1 \le n \le 5 \cdot 10^5, 1 \le l \le r \le 10^{17})$.

The second line contains n integers a_i $(1 \le a_i \le r)$ — the description of the sequence.

Output

Output a single integer — the maximum length of such a subsequence b.

Scoring

1. (1 point): all a_i are the same;
2. (3 points): $a_i = a_{i+2}$ for all $1 \le i \le n-2$;
3. (9 points): $n \le 20$;
4. (8 points): $n \le 5000$;
5. (9 points): $r - l \le 10$;
6. (10 points): $l = 1, r \le 10^6$;
7. (13 points): $r \le 10^6$;
8. (10 points): $l = 1$;
9. (24 points): $n \le 10^5$;
10. (13 points): no additional constraints.

Examples

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

Note

In the first example, you can select the subsequence $[1, 3, 2]$. $2 \le 1 + 3 \le 6$. $2 \le 3 + 2 \le 6$.

You can also select $[1, 4, 2]$.

Problem E. Matches

Time limit: 2 seconds
Memory limit: 256 megabytes

Anton has invented a new team sport called waterball (similar to paintball, but with water). He wants to share his invention with his n friends. Anton has good relationships with all his friends, but it's not guaranteed that his friends have the same relationships with each other. Specifically, we know that friend number a_i conflicts with friend b_i.

Anton was given a list of m conflicting pairs $(a_i; b_i)$. Now, it seems like it would be possible to divide the players into two teams, but it's not that simple for Anton...

He wants to divide these m conflicting pairs into segments in such a way that:

1. each conflicting pair belongs to exactly one segment;
2. considering only the relationships within each segment, it should be possible to divide all the people into two teams in such a way that there are no two conflicting people in the same team.

For example, let's say we have an array of conflicting pairs $[(1, 2), (2, 3), (1, 3)]$. We can take the first two pairs in the first segment. In this case, we can form the teams: $[1, 3]$ and $[2]$. In the second segment, we can take the last pair. 1 and 3 must be in different teams, and 2 can be in either team. Alternatively, we can assign the first pair to the first segment, and the last two to the second segment. Note that we cannot assign the first and third pairs to the same segment, and the second pair to another. This is because a segment should only contain consecutive pairs. We also cannot assign all pairs to the same segment, as there would always be a team where people conflict.

Anton has made the statement complicated again, and now he can't solve the problem. Help him by telling him the minimum number of segments into which he can divide the pairs to satisfy the above conditions.

Input

The first line contains two integers n, m ($1 \le n, m \le 10^6$) — the number of friends and the number of conflicting relationships among friends.

The next m lines contain two integers each, a_i, b_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$), indicating that friend a_i conflicts with friend b_i.

It is guaranteed that no pair $(a; b)$ is repeated more than once.

Output

Print a single integer — the answer to the problem.

Scoring

1. (4 points): $n \le 3$;
2. (7 points): $n \le 10$;
3. (15 points): $n, m \le 5000$;
4. (13 points): conflicting pairs in the input are randomly generated; this means that m pairs were randomly selected from all $\frac{n(n-1)}{2}$ pairs;
5. (14 points): each person conflicts with no more than 10 people;
6. (19 points): $n \le 10^5$;
7. (17 points): $n \le 2 \cdot 10^5$;

8. (11 points): without additional restrictions.

Examples

standard input	standard output
3 3 1 2 2 3 1 3	2
5 10 2 4 1 2 3 4 1 3 1 5 4 5 2 3 3 5 1 4 2 5	3

Note

The first example is explained in the legend above.

In the second example, for instance, it is possible to divide into the following segments: $[1; 6]$, $[7; 9]$, $[10; 10]$.

In the first segment, the teams can be formed as $[1, 4]$, $[2, 3, 5]$ — 1 and 4 do not conflict with each other, as well as the pairs $(2; 3)$, $(2; 5)$, $(3; 5)$.

In the second segment, the teams can be formed as $[1, 3]$, $[2, 4, 5]$ — 1 and 3 do not conflict with each other, as well as the pairs $(2; 4)$, $(2; 5)$, $(4; 5)$.

In the third segment, the teams can be formed as $[1, 2]$, $[3, 4, 5]$ — 1 and 2 do not conflict with each other, as well as the pairs $(3; 4)$, $(3; 5)$, $(4; 5)$.