Educational Codeforces Round 41 (Rated for Div. 2)

A. Tetris

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

You are given a following process.

There is a platform with $$$n$$$ columns. $$$1 \times 1$$$ squares are appearing one after another in some columns on this platform. If there are no squares in the column, a square will occupy the bottom row. Otherwise a square will appear at the top of the highest square of this column.

When all of the $$$n$$$ columns have at least one square in them, the bottom row is being removed. You will receive $$$1$$$ point for this, and all the squares left will fall down one row.

You task is to calculate the amount of points you will receive.

Input
The first line of input contain 2 integer numbers $$$n$$$ and $$$m$$$ ($$$1 \le n, m \le 1000$$$) — the length of the platform and the number of the squares.

The next line contain $$$m$$$ integer numbers $$$c_1, c_2, \dots, c_m$$$ ($$$1 \le c_i \le n$$$) — column in which $$$i$$$-th square will appear.

Output
Print one integer — the amount of points you will receive.

Example

input
3 9 1 1 2 2 2 3 1 2 3
output
2

Note
In the sample case the answer will be equal to $$$2$$$ because after the appearing of $$$6$$$-th square will be removed one row (counts of the squares on the platform will look like $$$[2~ 3~ 1]$$$, and after removing one row will be $$$[1~ 2~ 0]$$$).

After the appearing of $$$9$$$-th square counts will be $$$[2~ 3~ 1]$$$, and after removing one row it will look like $$$[1~ 2~ 0]$$$.

So the answer will be equal to $$$2$$$.

B. Lecture Sleep

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

Your friend Mishka and you attend a calculus lecture. Lecture lasts n minutes. Lecturer tells a_i theorems during the i-th minute.

Mishka is really interested in calculus, though it is so hard to stay awake for all the time of lecture. You are given an array t of Mishka's behavior. If Mishka is asleep during the i-th minute of the lecture then t_i will be equal to 0, otherwise it will be equal to 1. When Mishka is awake he writes down all the theorems he is being told — a_i during the i-th minute. Otherwise he writes nothing.

You know some secret technique to keep Mishka awake for k minutes straight. However you can use it **only once**. You can start using it at the beginning of any minute between 1 and $n - k + 1$. If you use it on some minute i then Mishka will be awake during minutes j such that $j \in [i, i + k - 1]$ and will write down all the theorems lecturer tells.

You task is to calculate the maximum number of theorems Mishka will be able to write down if you use your technique **only once** to wake him up.

Input
The first line of the input contains two integer numbers n and k ($1 \le k \le n \le 10^5$) — the duration of the lecture in minutes and the number of minutes you can keep Mishka awake.

The second line of the input contains n integer numbers $a_1, a_2, \dots a_n$ ($1 \le a_i \le 10^4$) — the number of theorems lecturer tells during the i-th minute.

The third line of the input contains n integer numbers $t_1, t_2, \ldots t_n$ $(0 \le t_i \le 1)$ — type of Mishka's behavior at the i-th minute of the lecture.

Output
Print only one integer — the maximum number of theorems Mishka will be able to write down if you use your technique **only once** to wake him up.

Example

input
6 3 1 3 5 2 5 4 1 1 0 1 0 0
output
16

Note
In the sample case the better way is to use the secret technique at the beginning of the third minute. Then the number of theorems Mishka will be able to write down will be equal to 16.

C. Chessboard

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

Magnus decided to play a classic chess game. Though what he saw in his locker shocked him! His favourite chessboard got broken into 4 pieces, each of size n by n, n is **always odd**. And what's even worse, some squares were of wrong color. j-th square of the i-th row of k-th piece of the board has color $a_{k,i,j}$; 1 being black and 0 being white.

Now Magnus wants to change color of some squares in such a way that he recolors minimum number of squares and obtained pieces form a valid chessboard. Every square has its color different to each of the neightbouring by side squares in a valid board. Its size should be $2n$ by $2n$. You are allowed to move pieces but **not allowed to rotate or flip them**.

Input
The first line contains **odd** integer n $(1 \le n \le 100)$ — the size of all pieces of the board.

Then 4 segments follow, each describes one piece of the board. Each consists of n lines of n characters; j-th one of i-th line is equal to `1` if the square is black initially and `0` otherwise. Segments are separated by an empty line.

Output
Print one number — minimum number of squares Magnus should recolor to be able to obtain a valid chessboard.

Examples

input
1 0 0 1 0
output
1

input
3 101 010 101 101 000 101 010 101 011 010 101 010
output
2

D. Pair Of Lines

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

You are given n points on Cartesian plane. Every point is a lattice point (i. e. both of its coordinates are integers), and all points are distinct.

You may draw two straight lines (not necessarily distinct). Is it possible to do this in such a way that every point lies on at least one of these lines?

Input

The first line contains one integer n ($1 \le n \le 10^5$) — the number of points you are given.

Then n lines follow, each line containing two integers x_i and y_i ($|x_i|, |y_i| \le 10^9$)— coordinates of i-th point. All n points are distinct.

Output

If it is possible to draw two straight lines in such a way that each of given points belongs to at least one of these lines, print `YES`. Otherwise, print `NO`.

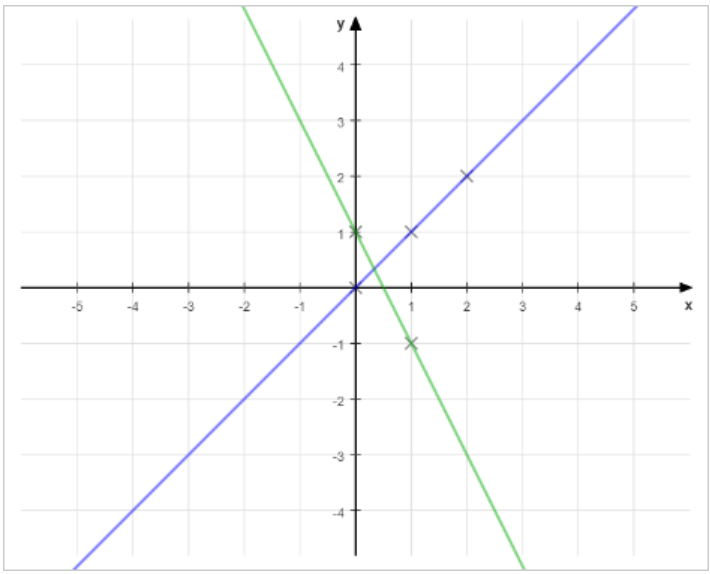
Examples

input
5 0 0 0 1 1 1 1 -1 2 2
output
YES

input
5 0 0 1 0 2 1 1 1 2 3
output
NO

Note

In the first example it is possible to draw two lines, the one containing the points 1, 3 and 5, and another one containing two remaining points.



E. Tufurama

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

One day Polycarp decided to rewatch his absolute favourite episode of well-known TV series "Tufurama". He was pretty surprised when he got results only for season 7 episode 3 with his search query of "Watch Tufurama season 3 episode 7 online full hd free". This got Polycarp confused — what if he decides to rewatch the entire series someday and won't be able to find the right episodes to watch? Polycarp now wants to count the

number of times he will be forced to search for an episode using some different method.

TV series have n seasons (numbered 1 through n), the i-th season has a_i episodes (numbered 1 through a_i). Polycarp thinks that if for some pair of integers x and y ($x < y$) exist both season x episode y and season y episode x then one of these search queries will include the wrong results. Help Polycarp to calculate the number of such pairs!

Input
The first line contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of seasons.

The second line contains n integers separated by space $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — number of episodes in each season.

Output
Print one integer — the number of pairs x and y ($x < y$) such that there exist both season x episode y and season y episode x.

Examples

input
5 1 2 3 4 5
output
0

input
3 8 12 7
output
3

input
3 3 2 1
output
2

Note
Possible pairs in the second example:

1. $x = 1, y = 2$ (season 1 episode 2 $\Leftrightarrow$ season 2 episode 1);
2. $x = 2, y = 3$ (season 2 episode 3 $\Leftrightarrow$ season 3 episode 2);
3. $x = 1, y = 3$ (season 1 episode 3 $\Leftrightarrow$ season 3 episode 1).

In the third example:

1. $x = 1, y = 2$ (season 1 episode 2 $\Leftrightarrow$ season 2 episode 1);
2. $x = 1, y = 3$ (season 1 episode 3 $\Leftrightarrow$ season 3 episode 1).

F. k-substrings

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a string s consisting of n lowercase Latin letters.

Let's denote k-substring of s as a string $subs_k = s_k s_{k+1} .. s_{n+1-k}$. Obviously, $subs_1 = s$, and there are exactly $\lceil \frac{n}{2} \rceil$ such substrings.

Let's call some string t an **odd proper suprefix** of a string T iff the following conditions are met:

- $|T| > |t|$;
- $|t|$ is an odd number;
- t is simultaneously a prefix and a suffix of T.

For evey k-substring ($k \in [1, \lceil \frac{n}{2} \rceil]$) of s you have to calculate the maximum length of its odd proper suprefix.

Input
The first line contains one integer n ($2 \le n \le 10^6$) — the length s.

The second line contains the string s consisting of n lowercase Latin letters.

Output

Print $\lceil \frac{n}{2} \rceil$ integers. i-th of them should be equal to maximum length of an odd proper suprefix of i-substring of s (or -1, if there is no such string that is an odd proper suprefix of i-substring).

Examples

input
15 bcabcabcabcabca
output
9 7 5 3 1 -1 -1 -1

input
24 abaaabaaaabaaabaaaabaaab
output
15 13 11 9 7 5 3 1 1 -1 -1 1

input
19 cabcabbcabcabbcabca
output
5 3 1 -1 -1 1 1 -1 -1 -1

Note

The answer for first sample test is folowing:

- 1-substring: bcabca**bcabcabca**
- 2-substring: cabcab**cabcabc**
- 3-substring: abcabc**abcab**
- 4-substring: bcabca**bca**
- 5-substring: cabcab**c**
- 6-substring: abcab
- 7-substring: bca
- 8-substring: c

G. Partitions

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

You are given a set of n elements indexed from 1 to n. The weight of i-th element is w_i. The weight of some subset of a given set is denoted as $W(S) = |S| \cdot \sum_{i \in S} w_i$. The weight of some partition R of a given set into k subsets is $W(R) = \sum_{S \in R} W(S)$ (recall that a partition of a given set is a set of its subsets such that every element of the given set belongs to exactly one subset in partition).

Calculate the sum of weights of all partitions of a given set into exactly k **non-empty** subsets, and print it modulo $10^9 + 7$. Two partitions are considered different iff there exist two elements x and y such that they belong to the same set in one of the partitions, and to different sets in another partition.

Input

The first line contains two integers n and k ($1 \le k \le n \le 2 \cdot 10^5$) — the number of elements and the number of subsets in each partition, respectively.

The second line contains n integers w_i ($1 \le w_i \le 10^9$)— weights of elements of the set.

Output

Print one integer — the sum of weights of all partitions of a given set into k **non-empty** subsets, taken modulo $10^9 + 7$.

Examples

input
4 2 2 3 2 3
output
160

input
5 2 1 2 3 4 5

output
645

Note

Possible partitions in the first sample:

1. $\{\{1, 2, 3\}, \{4\}\}$, $W(R) = 3 \cdot (w_1 + w_2 + w_3) + 1 \cdot w_4 = 24$;
2. $\{\{1, 2, 4\}, \{3\}\}$, $W(R) = 26$;
3. $\{\{1, 3, 4\}, \{2\}\}$, $W(R) = 24$;
4. $\{\{1, 2\}, \{3, 4\}\}$, $W(R) = 2 \cdot (w_1 + w_2) + 2 \cdot (w_3 + w_4) = 20$;
5. $\{\{1, 3\}, \{2, 4\}\}$, $W(R) = 20$;
6. $\{\{1, 4\}, \{2, 3\}\}$, $W(R) = 20$;
7. $\{\{1\}, \{2, 3, 4\}\}$, $W(R) = 26$;

Possible partitions in the second sample:

1. $\{\{1, 2, 3, 4\}, \{5\}\}$, $W(R) = 45$;
2. $\{\{1, 2, 3, 5\}, \{4\}\}$, $W(R) = 48$;
3. $\{\{1, 2, 4, 5\}, \{3\}\}$, $W(R) = 51$;
4. $\{\{1, 3, 4, 5\}, \{2\}\}$, $W(R) = 54$;
5. $\{\{2, 3, 4, 5\}, \{1\}\}$, $W(R) = 57$;
6. $\{\{1, 2, 3\}, \{4, 5\}\}$, $W(R) = 36$;
7. $\{\{1, 2, 4\}, \{3, 5\}\}$, $W(R) = 37$;
8. $\{\{1, 2, 5\}, \{3, 4\}\}$, $W(R) = 38$;
9. $\{\{1, 3, 4\}, \{2, 5\}\}$, $W(R) = 38$;
10. $\{\{1, 3, 5\}, \{2, 4\}\}$, $W(R) = 39$;
11. $\{\{1, 4, 5\}, \{2, 3\}\}$, $W(R) = 40$;
12. $\{\{2, 3, 4\}, \{1, 5\}\}$, $W(R) = 39$;
13. $\{\{2, 3, 5\}, \{1, 4\}\}$, $W(R) = 40$;
14. $\{\{2, 4, 5\}, \{1, 3\}\}$, $W(R) = 41$;
15. $\{\{3, 4, 5\}, \{1, 2\}\}$, $W(R) = 42$.