The 2026 ICPC China Wuhan National Invitational and Hubei Provincial Programming Contest

Contest Session

May 17, 2026

Problem List

A	Sort
B	Sequence Operations
C	Believe in You
D	Prime Game
E	Rook
F	Lottery
G	I Will Always Remember You
H	Rectangle Cutting
I	Nailoong vs. Bombloong 2
J	The Best Card
K	Deletion Game
L	String Matching
M	Iroha and the Kingdom of Construction

This problem set should contain 13 (thirteen) problems on 22 (twenty-two) numbered pages.

Please inform a runner immediately if something is missing from your problem set.

Problem A. Sort

Input file: `standard input`
Output file: `standard output`

Given a permutation p of length n.

Each operation randomly chooses an integer i with equal probability ($1 \leq i \leq n$), and then sorts the prefix interval $[1, i]$ and the suffix interval $[i+1, n]$ of the permutation p in ascending order, respectively. In particular, when $i = n$, the suffix interval is empty, which is equivalent to sorting the entire permutation.

You need to find the expected number of operations required until the final permutation becomes completely sorted in ascending order (that is, $p_j = j$).

The answer should be taken modulo 998244353.

Input

This problem contains multiple test cases.

The first line contains an integer T ($1 \leq T \leq 100$), the number of test cases.

For each test case:

- The first line contains an integer n ($1 \leq n \leq 500$), the length of the permutation.
- The second line contains n integers $p_1, p_2, \cdots, p_n$ ($1 \leq p_j \leq n$), representing the given permutation p. It is guaranteed that the given sequence is a permutation of $1 \sim n$.

It is guaranteed that the sum of n over all test cases does not exceed 2000.

Output

For each test case, output one line containing an integer, the expected number of operations modulo 998244353.

Example

standard input	standard output
3 3 3 1 2 2 2 1 5 3 2 5 1 4	748683267 2 670695427

Problem B. Sequence Operations

Input file: `standard input`
Output file: `standard output`

Given a sequence of n non-negative integers $a_1, a_2, \cdots, a_n$, and two sequences of length m, $c_1, c_2, \cdots, c_m$ and $x_1, x_2, \cdots, x_m$. **It is guaranteed that all x_i are positive integers.**

You need to perform m operations in order from 1 to m. In the i-th operation, you modify the current sequence a based on the value of c_i:

- If $c_i = 1$, you must perform $a_j \leftarrow \text{mex}(a_j, x_i)$ for all a_j.
- If $c_i = 2$, you must perform $a_j \leftarrow \gcd(a_j, x_i)$ for all a_j.
- If $c_i = 0$, you can **freely choose** to perform either of the two operations mentioned above.

After all m operations are completed, determine whether it is possible to make all elements in sequence a equal.

Additional notes regarding the mex and gcd operations are as follows:

- The binary operation $\text{mex}(u, v)$ is defined as the smallest non-negative integer that is not equal to u and not equal to v. For example: $\text{mex}(0, 1) = 2$, $\text{mex}(2, 2) = 0$.
- For the greatest common divisor gcd, it is specifically defined that $\gcd(0, x) = x$.

Input

The input contains multiple test cases. The first line contains an integer T ($1 \le T \le 10^4$), the number of test cases.

For each test case:

- The first line contains two integers n, m ($1 \le n, m \le 3 \times 10^5$), representing the length of sequence a and the number of operations.
- The second line contains n non-negative integers $a_1, a_2, \cdots, a_n$ ($0 \le a_j \le 10^9$), representing the initial sequence.
- The next m lines each contain two integers c_i, x_i ($0 \le c_i \le 2$, $1 \le x_i \le 10^9$), representing the type parameter and value parameter of the i-th operation.

It is guaranteed that across all test cases, $\sum n \le 3 \times 10^5$ and $\sum m \le 3 \times 10^5$.

Output

For each test case, output one line. If it is possible to make all numbers equal after all operations are finished, output "`Yes`"; otherwise, output "`No`".

Example

standard input	standard output
7 8 6 3 1 4 1 5 9 2 6 0 1 0 1 0 4 0 5 0 1 0 4 3 1 0 1 2 1 1 3 1 0 1 2 2 1 4 2 3 3 5 0 1 1 2 3 2 2 3 0 2 4 2 2 4 2 3 0 2 0 1 1 1 3 3 4 1 4 0 1 1 1 1 0 4 2 2	Yes No Yes No No No Yes

Note

For the first test case: in the first operation, choose to apply the gcd operation, and the sequence becomes $[1, 1, 1, 1, 1, 1, 1, 1]$. In the following 5 operations, no matter which operation you choose, the elements of the sequence will always remain equal, so output "`Yes`".

For the second test case: the only operation is $c_1 = 1$, $x_1 = 1$, so you can only choose the mex operation.

- $a_1 \leftarrow \text{mex}(0, 1) = 2$
- $a_2 \leftarrow \text{mex}(1, 1) = 0$
- $a_3 \leftarrow \text{mex}(2, 1) = 0$

The final sequence becomes $[2, 0, 0]$, and it is impossible to make all numbers equal, so output "`No`".

Problem C. Believe in You

Input file: `standard input`
Output file: `standard output`

Slay the Spire 2 is a world-renowned game. In this game, you play as a character climbing a high tower, using cards to defeat monsters along the way until victory is achieved.

Now you must face a powerful enemy. Unfortunately, your remaining health cannot withstand even a single unblocked attack. Fortunately, you know the cards you will draw and the monster's actions for the next n turns. You want to quickly determine whether you can defeat the monster within n turns without dying.

The cards in the game are divided into two types:

- Strike: Playing this card deals S points of damage.
- Defend: Playing this card provides D points of block.

The condition to defeat the enemy is that the **total damage** you deal is greater than or equal to the monster's health HP.

In the i-th turn, the player's actions are defined as:

- Discard the cards currently in hand and clear any remaining block from the previous turn (if any).
- Draw a_i Strike cards and $5 - a_i$ Defend cards.
- Choose and play no more than 3 cards; each card can only be played once.
- End the action.

In the i-th turn, the monster's action is defined as:

- Deal k_i points of damage to the player. If k_i is greater than the player's current block value, you will die, and the game ends immediately.
- End the action.

In each turn, the order of actions for both parties is:

1. Player's turn.
2. If the monster dies, the player wins, and the game ends.
3. Monster's turn.
4. If the player dies, the game ends.

You need to determine if you can defeat the monster within n turns. If you can, output "`Yes`" and the earliest turn in which the monster can be defeated; if you cannot defeat the monster within n turns or if the player dies during the process, output "`No`".

Input

The first line contains an integer n ($1 \le n \le 100$), representing the predicted number of game turns.

The second line contains three positive integers S, D, HP ($1 \le S, D \le 10$, $1 \le HP \le 100$), representing the damage dealt by a Strike, the block provided by a Defend, and the monster's health, respectively.

The next n lines each contain two non-negative integers a_i, k_i ($0 \le a_i \le 5$, $0 \le k_i \le 50$), describing the card draw situation for the i-th turn (drawing a_i Strikes) and the monster's attack damage.

Output

If the monster can be defeated within n turns, output "Yes" on one line, and output an integer on the next line representing the earliest turn in which the monster is defeated.

If the monster cannot be defeated within n turns or if the player dies, output "No".

Examples

standard input	standard output
4 5 5 25 4 0 1 10 2 12 2 12	Yes 3
5 1 5 4 1 4 2 0 5 3 0 4 0 1	Yes 3
4 1 5 5 0 1 3 3 0 5 2 1	No
5 1 4 4 5 4 1 5 3 1 4 4 5 4	No
4 4 5 4 1 2 1 0 4 5 1 1	Yes 1

Note

For the 1st sample case, the optimal card-playing strategy is as follows:

- Turn 1: The player draws 4 Strikes and 1 Defend. The monster's attack power is 0. The player plays 3 Strikes, dealing 15 points of damage. At the end of the turn, the monster has 15 health remaining. The player receives 0 damage and survives.
- Turn 2: The player draws 1 Strike and 4 Defends. The monster's attack power is 10. To survive, the player must play 2 Defends to gain 10 block, leaving 1 remaining card-play action to play 1 Strike, dealing 5 points of damage. At the end of the turn, the monster has 10 health remaining. The player receives a 10-point attack, which is negated by the block, and survives.

- Turn 3: The player draws 2 Strikes and 3 Defends. Since the monster will not perform its attack phase if it dies, the player can directly play 2 Strikes, dealing 10 points of damage. At this point, the cumulative damage reaches 30, the monster dies, and the game ends in the 3rd turn.

Problem D. Prime Game

Input file: `standard input`
Output file: `standard output`

Iroha and Kaguya have recently become fascinated by a mathematical game based on integers.

Initially, they share a positive integer n. The two players take turns performing operations; a single round is defined as both Iroha and Kaguya completing one operation each. In each round:

1. Iroha goes first: She must choose a non-negative integer x ($0 \le x < n$) and replace the current n with $n + x$.

2. Kaguya goes second: She must choose a **prime number** p that divides the current n (i.e., $p \mid n$) and replace the current n with $\frac{n}{p}$.

The above operations repeat continuously. If $n = 1$ after the end of any round, the game ends immediately.

During the game, Iroha aims to maximize the total number of rounds played, while Kaguya aims to minimize the total number of rounds. Assuming both players are perfectly rational and adopt optimal strategies, please determine the total number of rounds the game will ultimately last.

Input

The first line contains an integer n ($1 \le n \le 10^{18}$), representing the initial number of the game.

Output

Output a single line containing an integer, representing the total number of rounds the game will last given that both players play optimally.

Examples

standard input	standard output
1	0
9	4
16	5

Problem E. Rook

Input file: `standard input`
Output file: `standard output`

There is an $N \times M$ chessboard.

You need to place K **distinct** rooks on this chessboard. The placement must satisfy all of the following conditions:

1. Each cell contains at most one rook.
2. Every row of the chessboard contains at least one rook.
3. Every column of the chessboard contains at least one rook.

Please find the total number of valid placements. Since the answer may be very large, you only need to output the result modulo 998244353.

Input

The input contains only one line.

This line contains three integers N, M, K in order ($1 \le N, M, K \le 10^5$), representing the number of rows and columns of the chessboard, and the number of rooks to place, respectively.

Output

Output one line containing an integer, the number of valid placements modulo 998244353.

Examples

standard input	standard output
2 2 3	24
5 3 4	0
353 442 899	623104742

Problem F. Lottery

Input file: `standard input`
Output file: `standard output`

A lottery event is being held in the Tsukuyomi.

There are n items in the prize pool, and their values are $A_1, A_2, \cdots, A_n$.

Kaguya can participate in the lottery for several rounds, but she must participate in at least one round. In each round, the process is as follows:

1. Kaguya needs to pay the current lottery cost c, and then an integer x is chosen uniformly at random from $[1, n]$. She learns which item x was drawn.
2. After seeing the result, Kaguya has two choices:
   - Keep the x-th item and end the entire lottery event. In this case, she will obtain the item with value A_x.
   - Give up the currently drawn item and enter the next round of the lottery. However, the lottery cost will increase accordingly, and the cost of the next round will become $c+k$, that is, update $c \leftarrow c+k$.

Kaguya defines the "profit" of this lottery event as the value of the finally obtained item A_x minus the sum of the lottery costs she paid in all rounds.

Assuming Kaguya is smart enough and always takes the optimal strategy to maximize her expected profit, please find the expected profit she can obtain under the optimal strategy.

Input

The first line contains three integers n, c, k ($1 \le n \le 4 \times 10^5$, $1 \le c \le 4 \times 10^5$, $0 \le k \le 4 \times 10^5$), representing the total number of items in the prize pool, the initial cost of the first lottery round, and the increase in cost for each reroll.

The second line contains n integers $A_1, A_2, \cdots, A_n$ ($1 \le A_i \le 4 \times 10^5$), representing the value of each item.

Output

Output one line containing a real number, representing the expected profit under Kaguya's optimal strategy. Your answer is considered correct if and only if the absolute or relative error between your answer and the standard answer does not exceed 10^{-6}.

Examples

standard input	standard output
3 2 0 4 1 6	2.000000000
9 8 2 53 71 5 6 80 37 50 33 90	51.557786840

Problem G. I Will Always Remember You

Input file: `standard input`
Output file: `standard output`

Given a directed acyclic graph G with n vertices and m edges, and a color sequence $a_1, a_2, \cdots, a_n$ of length n, where a_i denotes the initial color of vertex i.

Let S_i denote the set of all vertices reachable from vertex i in graph G (including vertex i itself).

Now, you need to process q operations in order. The operations are of the following two types:

- Given x and y, change the color a_x of vertex x to y.
- Given x, query the number of distinct colors among all vertices reachable from vertex x. That is, find the size of the set $\{a_j \mid j \in S_x\}$.

Input

The first line contains two integers n, m ($1 \leq n \leq 1.5 \times 10^5$, $0 \leq m \leq 3 \times 10^5$), representing the number of vertices and the number of edges in the graph, respectively.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \leq a_i \leq n$), representing the initial color of each vertex.

The next m lines each contain two integers u, v ($1 \leq u, v \leq n$), indicating that there is a directed edge from u to v in graph G. It is guaranteed that graph G is a directed acyclic graph (DAG) and contains no multiple edges.

The next line contains an integer q ($1 \leq q \leq 1.5 \times 10^5$), representing the total number of operations.

The next q lines each describe an operation in one of the following two formats:

- `1 x y`: Represents an update operation, changing the color a_x to y ($1 \leq x, y \leq n$).
- `2 x`: Represents a query operation, querying the number of distinct colors reachable from vertex x ($1 \leq x \leq n$).

Output

For each operation of the second type, output a single line containing an integer representing the answer to the query.

Example

standard input	standard output
9 14 8 4 2 3 5 2 2 4 7 3 5 8 9 4 5 5 8 6 8 3 8 1 4 2 9 2 4 2 8 1 3 7 9 1 8 5 7 10 1 6 4 1 3 3 2 1 2 3 1 4 3 1 8 5 1 8 7 2 7 2 2 2 9	6 5 2 5 1

Problem H. Rectangle Cutting

Input file: `standard input`
Output file: `standard output`

Yachiyo has a rectangle in the first quadrant of the Cartesian coordinate plane. The lower-left corner of the rectangle is at the origin $(0, 0)$, and the upper-right corner is at (n, m). In other words, the two sides of the rectangle are aligned with the x-axis and the y-axis, with lengths n and m along the x- and y-directions, respectively.

Yachiyo decides to make q cuts on this rectangle. Specifically, there are two kinds of cuts:

- Horizontal cut: given a positive integer k, cut along the line $y = k$.
- Vertical cut: given a positive integer k, cut along the line $x = k$.

Each cut goes through the entire region, further dividing the existing rectangles into more smaller rectangles.

Yachiyo wants to know, after each cut is completed, what the maximum area among all the currently divided small rectangles is.

Input

The first line contains three integers n, m, q ($1 \le n, m \le 10^9$, $1 \le q \le 5 \times 10^5$), representing the horizontal length, vertical length, and the number of cuts of the initial rectangle, respectively.

The next q lines each contain two integers op, k, describing one cut operation:

- If $op = 1$, it means a vertical cut is performed, and the given integer k ($1 \le k < n$) indicates cutting along the line $x = k$.
- If $op = 2$, it means a horizontal cut is performed, and the given integer k ($1 \le k < m$) indicates cutting along the line $y = k$.

It is guaranteed that Yachiyo will never cut at the same position twice (that is, all cutting lines are distinct).

Output

Output q lines. The i-th line should contain one integer, representing the maximum area among all small rectangles after the i-th cut.

Example

standard input	standard output
5 3 3 1 1 1 4 2 2	12 9 6

Note

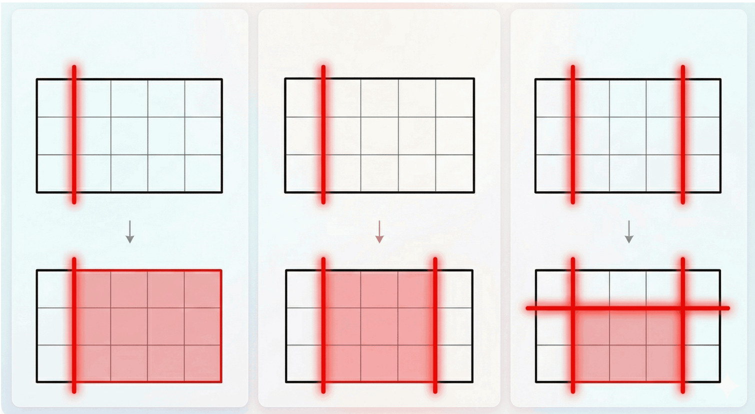

Рис. 1: *Sample explanation*

Problem I. Nailoong vs. Bombloong 2

Input file: `standard input`
Output file: `standard output`

This is a communication problem. In this problem, your program will be run twice. Between the two runs, all variables stored in memory will be lost, but the information obtained in the first run may be crucial for correctly solving the problem in the second run.

There are two characters in this problem: "Nailoong" and "Bombloong".

Nailoong has the complete structural information of a tree with n nodes, while Bombloong only knows the number of nodes n. Since Bombloong has been imprisoned by the evil Little Leopard, Nailoong can only help Bombloong reconstruct a tree that is **isomorphic** to the original tree through a special one-way communication method.

The rules of communication are as follows:

- Initially, all nodes in the tree are white.
- Nailoong can perform $m \le n-3$ operations (the value of m is determined by Nailoong).
- In the i-th operation, Nailoong needs to choose a node x_i on the tree and flip its color (white to black, black to white).
- After each flip, let a_i be the number of edges in the current tree whose two endpoints have different colors.
- After m operations, Nailoong obtains a sequence $a_1, a_2, \cdots, a_m$ of length m.

Nailoong cannot directly send node indices to Bombloong; she can only send the total number of operations m and the sequence a to Bombloong. After receiving n, m, and the sequence a, Bombloong needs to construct and output a tree that is **isomorphic** to Nailoong's tree.

Communication Protocol

In each test case, the contestant's program will be run twice. A testing tool is provided in the handout files for local debugging.

First Run

In the first run, you will play the role of "Nailoong".

Input

The first line of input contains an integer 1, which allows your program to identify that this is the first run.

The second line contains an integer n ($4 \le n \le 3 \times 10^5$), representing the number of nodes in the tree.

The next $n-1$ lines each contain two integers u, v ($1 \le u, v \le n$), representing an edge in the tree.

Output

The first line of output should contain an integer m ($0 \le m \le n-3$), representing the number of operations.

The second line contains m integers $x_1, x_2, \cdots, x_m$ ($1 \le x_i \le n$), representing the indices of the nodes flipped by Nailoong in each operation in sequence.

Second Run

In the second run, you will play the role of "Bombloong".

Input

The first line of input contains an integer 2, which allows your program to identify that this is the second run.

The second line contains two integers n, m ($4 \le n \le 3 \times 10^5$, $0 \le m \le n - 3$), representing the number of nodes and the number of operations performed by Nailoong, respectively.

If $m > 0$, the third line contains m integers $a_1, a_2, \cdots, a_m$ ($0 \le a_i \le n$), representing the sequence generated by the judge based on Nailoong's operations; if $m = 0$, the third line does not exist.

Output

Output $n - 1$ lines, each containing two integers u, v, representing an edge of the tree you reconstructed ($1 \le u, v \le n$). You must ensure that the output tree is isomorphic to the tree provided in the input of the first run.

Examples

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>1
4
1 2
2 3
3 4</pre></td><td><pre>1
2</pre></td></tr>
<tr><td><pre>2
4 1
2</pre></td><td><pre>1 3
3 2
2 4</pre></td></tr>
</table>

Note

The two samples demonstrate the two runs within the same test case.

Nailoong flipped node 2. At this point, the number of edges with different colored endpoints is 2, so $a_1 = 2$.

Bombloong received the sequence $a = [2]$. Since Bombloong and Nailoong share a deep connection, she immediately guessed the structure of the tree. Although the tree obtained by Bombloong differs from the tree Nailoong held in terms of node indexing, they are isomorphic, so it is still considered correct.

Problem J. The Best Card

Input file: `standard input`
Output file: `standard output`

Snakebite is the most powerful card in the game *Slay the Spire 2*, and Yuki enjoys using it to defeat enemies.

The game lasts for 10^{1000} turns. At the start of the game, the enemy has 0 layers of poison.

Yuki has n snakebite cards in total. The effect of the i-th snakebite is: when played, it increases the enemy's poison layers by v_i. Yuki will play the i-th snakebite at the start of the t_i-th turn.

At the end of each turn, let x be the number of poison layers on the enemy:

- If $x = 0$, nothing happens;
- If $x \neq 0$, the enemy takes x damage, and the number of poison layers decreases by 1, i.e., $x \leftarrow x - 1$.

To maximize the total damage dealt to the enemy, Yuki can perform k enchantments before the game begins. In each enchantment, Yuki chooses a positive integer $i \leq n$ and increases the value of v_i by 1. The same index i can be selected multiple times.

You need to determine the maximum damage the enemy can receive after Yuki performs k enchantments.

Input

Each test contains multiple test cases.

The first line contains an integer t ($1 \leq t \leq 10^4$), representing the number of test cases.

For each test case:

- The first line contains two integers n, k ($1 \leq n \leq 3 \cdot 10^5$, $0 \leq k \leq 10^9$).
- The second line contains n integers $v_1, v_2, \ldots, v_n$ ($1 \leq v_i \leq 10^9$, $1 \leq \sum v_i \leq 10^9$).
- The third line contains n integers $t_1, t_2, \ldots, t_n$ ($1 \leq t_i \leq 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed $3 \cdot 10^5$.

Output

For each test case, output a single line containing an integer representing the maximum damage the enemy can receive after Yuki performs k enchantments.

Example

standard input	standard output
5 2 1 2 3 1 4 2 2 3 4 1 3 4 1 3 3 1 4 4 4 1 5 3 0 16 27 62 9 88 11 3 5 9 3 1 8 3 3	13 37 63 3335 126

Problem K. Deletion Game

Input file: `standard input`
Output file: `standard output`

Yachiyo has an integer sequence S of length n, and each position in the sequence (indexed from 1) has a weight a_i.

She can copy this sequence together with its weights and concatenate the copies end to end several times. Specifically, if she chooses to concatenate m copies in total ($m \geq 1$), she will obtain a new sequence S' of length $m \times n$ and a corresponding new weight sequence a'. For any $0 \leq c < m$ and $1 \leq i \leq n$, we have $S'_{cn+i} = S_i$ and $a'_{cn+i} = a_i$.

Now Yachiyo wants to perform several elimination operations on the resulting sequence S' (possibly zero times). Each operation works as follows:

- Choose two indices i, j such that $1 \leq i < j \leq |S'|$ and $S'_i = S'_j$.
- Delete the $(i+1)$-th through j-th elements from both S' and a'.
- After the operation, the remaining elements are compacted in their original order, and the total length decreases accordingly.

Yachiyo wants to know: after choosing an appropriate number of copies and performing any number of elimination operations, what is the minimum possible sum of the weights of the remaining sequence? Also, under the condition that this minimum weight sum is achieved, what is the minimum number of copies of the original sequence needed (that is, the minimum m)?

Input

The input contains three lines.

The first line contains an integer n ($1 \leq n \leq 3 \times 10^5$), the length of the initial sequence S.

The second line contains n integers $S_1, S_2, \cdots, S_n$ ($1 \leq S_i \leq 3 \times 10^5$), representing the elements of the initial sequence S.

The third line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \leq a_i \leq 3 \times 10^5$), representing the weight of each position.

Output

Output one line containing two integers separated by a space:

- The first integer is the minimum possible sum of weights of the remaining sequence after the operations.
- The second integer is the minimum total number of copies of the original sequence needed to achieve that minimum weight sum.

Example

standard input	standard output
6 1 1 4 5 1 4 1 9 1 9 8 10	2 1

Note

In this sample, the optimal strategy is to use only 1 copy of the original sequence (that is, $m = 1$, with no extra copying).

The initial sequence S' is $[1, 1, 4, 5, 1, 4]$, and the corresponding weight sequence a' is $[1, 9, 1, 9, 8, 10]$.

Yachiyo can perform the following two operations:

1. Choose $i = 1, j = 2$ (at this time $S'_1 = S'_2 = 1$), and delete the $(i + 1)$-th through j-th elements (that is, delete the 2nd element). After the operation, S' becomes $[1, 4, 5, 1, 4]$, and a' becomes $[1, 1, 9, 8, 10]$.
2. In the new sequence, choose $i = 2, j = 5$ (at this time $S'_2 = S'_5 = 4$), and delete the $(i+1)$-th through j-th elements (that is, delete the 3rd, 4th, and 5th elements). After the operation, the remaining S' is $[1, 4]$, and the remaining a' is $[1, 1]$.

At this point, no further elimination is possible, and the sum of the remaining weights is $1 + 1 = 2$. It can be proved that under any number of copies and any sequence of operations, the weight sum cannot be smaller than 2.

Problem L. String Matching

Input file: `standard input`
Output file: `standard output`

Given n strings $s_1, s_2, \cdots, s_n$ consisting of lowercase letters. It is guaranteed that these strings are sorted in **non-decreasing** order of length, i.e. $|s_1| \le |s_2| \le \cdots \le |s_n|$.

For any two different strings s_i and s_j with $1 \le i < j \le n$, define $t_{i,j} = s_i + s_j$, which means the new string obtained by concatenating s_i and s_j in order, with s_i in front and s_j behind.

We define a function $f(t)$, where t is a string of length m. The value of $f(t)$ is the number of positive integers x $(0 < x \le m)$ satisfying the following condition:

- The first x characters of string t are exactly the same as the last x characters of t. That is, $t[1 : x] = t[m - x + 1 : m]$.

Your task is to compute the sum of $f(t_{i,j})$ over all valid pairs (i, j), that is:

$$\sum_{1 \le i < j \le n} f(t_{i,j}).$$

Input

The input contains multiple lines.

The first line contains an integer n $(1 \le n \le 10^6)$, representing the total number of given strings.

The next n lines each contain a lowercase string s_i $(1 \le |s_i| \le 10^6)$.

It is guaranteed that the total length of all strings in the test data satisfies $\sum |s_i| \le 10^6$.

Output

Output one line containing an integer, representing the total sum of all $f(t_{i,j})$.

Examples

standard input	standard output
2 ak hbcpc	1
3 aa abaa ababaa	9

Note

In the second sample, there are $n = 3$ strings, and we can form 3 different $t_{i,j}$:

- When $i = 1, j = 2$: $t_{1,2} =$ aaabaa. The lengths x such that the prefix equals the suffix are: 1, 2, 6.
- When $i = 1, j = 3$: $t_{1,3} =$ aaababaa. The lengths x such that the prefix equals the suffix are: 1, 2, 8.
- When $i = 2, j = 3$: $t_{2,3} =$ abaaababaa. The lengths x such that the prefix equals the suffix are: 1, 4, 10.

The final answer is $3 + 3 + 3 = 9$.

Problem M. Iroha and the Kingdom of Construction

Input file:	`standard input`
Output file:	`standard output`

After wandering around for quite a while, Iroha arrived at a mysterious place in the Moon Reading Space — the Kingdom of Construction.

Having heard that you had once been here, Iroha was very happy and shared with you an interesting memory from the Kingdom of Construction.

In the Kingdom of Construction, there is a magical deck of cards. It has exactly n different ranks (represented by integers from 1 to n), and each rank appears exactly twice. After a uniform shuffle, these $2n$ cards form a deck. **In other words: the input data guarantees that the deck is generated uniformly at random from all valid decks containing n ranks, with exactly two cards of each rank.**

In this space, Iroha is granted a very powerful ability: generating stacks. Since this ability is very exhausting, Iroha can generate only 260 stacks.

Now, Iroha needs to take the cards from the original deck one by one in order from top to bottom. For each card taken out, Iroha may put it onto the top of any stack.

After all $2n$ cards have been placed, Iroha's stacks must satisfy the following condition: it must be possible to perform several "elimination operations" until all cards are eliminated:

- Each operation chooses **two different stacks**. If the top cards of these two stacks have the same rank, then these two cards are popped (removed) simultaneously.

Unfortunately, Iroha could not solve this problem, so she handed it over to the smart and experienced you, hoping that you could help her.

To make it easier for you, you do not need to give the actual elimination process. You only need to output, for each card in the original deck order, which stack it is placed onto. Iroha will help you check whether your construction is correct.

Input

The first line contains a positive integer n ($1 \le n \le 5 \times 10^5$), denoting the number of card ranks.

The second line contains $2n$ positive integers $a_1, a_2, \ldots, a_{2n}$ ($1 \le a_i \le n$), representing the ranks of the cards in the original deck from top to bottom. It is guaranteed that each rank in $\{1, 2, \ldots, n\}$ appears exactly twice.

It is guaranteed that there are exactly 50 tests (including the sample), and in each test, the array a is chosen uniformly at random from all sequences of length $2n$ in which each rank in $\{1, 2, \ldots, n\}$ appears exactly twice.

Output

Output one line containing $2n$ positive integers $c_1, c_2, \ldots, c_{2n}$, where $1 \le c_i \le 260$, indicating that the i-th card taken from the original deck is put into the c_i-th stack.

If there are multiple valid solutions, output any of them.

Example

standard input	standard output
3 1 2 3 1 3 2	1 1 1 2 2 3

Note

After processing your command, the last 257 stacks are empty, and the first 3 stacks are as follows (elements are listed from bottom to top):

1. $[1, 2, 3]$,
2. $[1, 2]$, and
3. $[3]$.

It is easy to prove that there exists a way to eliminate all cards.