CSE221 Assignment 03 Summer 2025

A. Count the Inversion

time limit per test: 1 second
memory limit per test: 256 megabytes

Here is a Pseudocode of the Merge Sort Algorithm.

```
def merge(a, b):
    # write your code here
    # a and b are two sorted list
    # merge function will return a sorted list after merging a and b

def mergeSort(arr):
    if len(arr) <= 1:
        return arr
    else:
        mid = len(arr)//2
        a1 = mergeSort(............)  # write the parameter
        a2 = mergeSort(............)  # write the parameter
        return merge(a1, a2)          # complete the merge function above
```

Now, you are given an array ***A*** of size ***N*** of ***N*** distinct integers. It is guaranteed that the array A contains a permutation of integers from 1 to N (i.e., every integer from 1 to N appears exactly once).

1. Count the number of inversions in the given array.
2. Sort the array in non-decreasing order.

An inversion is a pair (i, j) where $i < j$ and $A[i] > A[j]$.

Input
The first line contains an integer ***N*** $(1 \le N \le 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \le a_i \le N)$ separated by spaces.

Output
In the first line, print the total number of inversions in the given array. In the next line, print the array in non-decreasing order.

Examples

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

input
```
5
1 2 3 4 5
```
output
```
0
1 2 3 4 5
```

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

input
```
7
6 4 2 5 7 3 1
```
output
```
14
1 2 3 4 5 6 7
```

Note
In the first example (1-based indexing), the inversions are the pairs of indices $(3,\ 4), (3,\ 5)$ and $(4,\ 5)$.

In the second example, there are no inversions.

In the third example, every pair of i, j where $i < j$, we have $A[i] > A[j]$. Hence, All 10 such pairs are inversions.

B. Count the Inversion Revisited

time limit per test: 1 second
memory limit per test: 256 megabytes

You are given an array A of N integers. Find the number of pairs of indexes (i, j) such that $i < j$ and $A[i] > A[j]^2$.

Input
The first line contains an integer N $(1 \le N \le 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(-10^6 \le a_i \le 10^6)$ separated by spaces.

Output
Output a single integer - number of such pairs.

Examples

input
```
6
10 2 5 1 -2 25
```
output
```
6
```

input
```
5
5 4 3 -2 -1
```
output
```
4
```

C. Fast Power Drift

time limit per test: 1 second
memory limit per test: 256 megabytes

You are given two integers ***a*** and ***b***. Calculate $a^b \ mod \ 107$.

Input
The input file contains two integers ***a*** $(1 \le a \le 10^4)$ and ***b*** $(1 \le b \le 10^{12})$.

Output
Print one integer — the result of $a^b \ mod \ 107$.

Examples

input
```
100 3
```
output
```
85
```

input
```
100 5
```
output
```
99
```

input
```
10000 1000000000000
```
output
```
27
```

D. Fast Matrix Drift

time limit per test: 1 second
memory limit per test: 256 megabytes

You are given a 2×2 integer matrix A and an integer exponent X $(1 \le X \le 10^9)$.

Your task is to compute the matrix A^X, where all intermediate operations (additions and multiplications during matrix multiplication) are performed modulo $10^9 + 7$.

Formally, for two matrices:

$$B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}, \quad C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}.$$

Their product $D = B \times C$ is defined as: $D_{ij} = (b_{i1} \cdot c_{1j} + b_{i2} \cdot c_{2j}) \bmod (10^9 + 7)$.

Input
The first line contains a single integer T $(1 \le T \le 10^5)$ — the number of test cases.

The first line of each test case contains four integers a_{11}, a_{12}, a_{21}, and a_{22} where $0 \le a_{ij} \le 10^9$ — the elements of the matrix A. The second line contains an integer X where $1 \le X \le 10^9$.

Output
For each test case, print two lines, each containing two integers: the resulting 2×2 matrix A^X. Each element should be printed modulo $10^9 + 7$.

Examples

input
```
2
1 1 1 0
5
2 0 0 2
3
```
output
```
8 5
5 3
8 0
0 8
```

input
```
3
1 1 1 1
10
1 2 3 4
12
1 2 1 2
5
```
output
```
512 512
512 512
138067399 201223170
301834755 439902154
81 162
81 162
```

E. Fast Series Drift

time limit per test: 2.5 seconds
memory limit per test: 256 megabytes

You are given three integers ***a***, ***n*** and ***m***. Calculate $(a^1 + a^2 + \ldots + a^n) \% m$.

Input
The first line contains an integer ***T*** $(1 \le T \le 10^5)$ — total numbers of test cases.

In each of the next T test cases, there are three integers ***a*** $(1 \le a \le 10^6)$, ***n*** $(1 \le n \le 10^{12})$ and $(1 \le m \le 10^9)$

Output
Print one integer — the result of $(a^1 + a^2 + \ldots + a^n) \% m$.

Example

input
```
3
2 5 1000
2 9 1000
1 100 30
```
output
```
62
22
10
```

F. Ordering Binary Tree

time limit per test: 1 second
memory limit per test: 256 megabytes

you are given an array ***A*** of size ***N*** in **increasing** order. Find an order of these N integers such that, if these integers are inserted into a Binary Search Tree (BST) one by one, the height of the resulting BST is minimized.

A Binary Search Tree is a binary tree in which each node has at most two children, referred to as the left and right child. For any node, all elements in the left subtree are smaller than the node's value, and all elements in the right subtree are greater than the node's value.

The height of a Binary Search Tree is defined as the maximum depth among all the nodes in the tree.

Note: All the elements in the array A are guaranteed to be unique. In other words, $A_i \ne A_j$ if $i \ne j$.

Input
The first line contains an integer ***N*** $(1 \le N \le 10^5)$ — denoting the length of the list.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \le a_i \le 10^9)$ in non-descending order separated by spaces.

Output
Output the order of the elements such that when inserted into a Binary Search Tree, the height of the tree is minimized. If there are multiple such orders then find any of them.

Example

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

G. 220 Trees

time limit per test: 1 second
memory limit per test: 256 megabytes

There is a Binary Tree with ***N*** nodes. You are given the in–order and pre-order traversals of the tree. Your task is to determine the post-order traversal of the tree.

Input
The first line contains an integer N $(1 \le N \le 1000)$ — the number of nodes in the binary tree.

In the next line, there will be N integers $a_1, a_2, a_3 \ldots a_n$ $(1 \le a_i \le N)$ separated by spaces – representing the in-order traversal of the tree.

The following line, there will be N integers $b_1, b_2, b_3 \ldots b_n$ $(1 \le b_i \le N)$ separated by spaces – representing the pre-order traversal of the tree.

Output
Print N space-separated integers representing the post-order traversal of the binary tree.

Example

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

H. 220 Trees Reassessed

time limit per test: 1 second
memory limit per test: 256 megabytes

There is a Binary Tree with ***N*** nodes. You are given the in-order and post-order traversals of the tree. Your task is to determine the pre-order traversal of the tree.

Input
The first line contains an integer N $(1 \le N \le 1000)$ — the number of nodes in the binary tree.

In the next line, there will be N distinct integers $a_1, a_2, a_3 \ldots a_n$ $(1 \le a_i \le N)$ separated by spaces – representing the in-order traversal of the tree.

The following line, there will be N distinct integers $b_1, b_2, b_3 \ldots b_n$ $(1 \le b_i \le N)$ separated by spaces – representing the post-order traversal of the tree.

Output
Print N space-separated integers representing the pre-order traversal of the binary tree.

Examples

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

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