Birla Institute of Technology and Science, Pilani
Hyderabad Campus

CS F211 Data Structures and Algorithms
Lab 7: Heaps, Binary Trees and Binary Search Trees

Allowed languages: **C, C++**

General Tips

- Indent your code appropriately and use proper variable names. These increase readability and writability of the code. Also, use comments wherever necessary.
- Use a proper IDE or text editors like Sublime Text or VSCode as they help to run and test your code on multiple test-cases easily. However, in the lab you will be using command line interface.
- Try to use functions as much as possible in your code. Functions increase reusability and the pass-by-value feature provides a significant help sometimes. Modularizing your code also helps you to debug efficiently.
- For each problem, implement using the language(s) mentioned in the Allowed Language(s) section.

Directions for Binary Tree Problems

- The asked time complexity excludes reading the input and printing the output.
- If the input for the problem is a binary tree, it will be provided as an array of integers. Assume 1-based indexing for the array.

  - The elements of the array will contain the values of the nodes.
  - The values of the nodes are guaranteed to be non-negative integers. The values may not be unique.
  - The 1-st element of the array will contain the value of the root node.
  - The $(2i)$-th and $(2i+1)$-th elements will contain the values of the left child and right child of the node corresponding to the i-th element, respectively.
  - If $(2i)$ or $(2i+1)$ is greater than the size of the array then the corresponding child node does not exist.
  - If an element of the array has value -1, then the corresponding node does not exist.

- It is recommended that you define a self-referential structure for one node of the binary tree (similar to the one mentioned below) and construct the binary tree using it from the given input. However, it is not necessary to do so.

```
struct TreeNode {
    int val;
    TreeNode *left;
    TreeNode *right;
    TreeNode() : val(0), left(nullptr), right(nullptr) {}
    TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
    TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
};
```

Problem A. Min Sum

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

Jilind is going to buy n items one by one. The price of the i-th item is Rs a_i. He has m discount tickets, and he can use any number of them when buying an item. Every discount ticket can only be used once.

If y discount tickets are used when buying an item priced Rs x, he can get the item for Rs $\lfloor \frac{x}{2^y} \rfloor$.

What is the minimum amount of money required to buy all the items?

Assume 1-based indexing.

Note:

The time complexity of your solution should be $O((n+m)logn)$. Solution of higher time complexity will not be considered.

Allowed Language(s): C, C++

Input

The first line of input contains two integers, n and m ($1 \leq n, m \leq 10^5$). m can be less than, equal to or greater than n.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i \leq 10^9$). The elements of a may not be unique.

Output

Print one line containing the minimum amount of money required to buy all the items.

Example

standard input	standard output
3 3 2 13 8	9

Explanation

In the given example, Jilind can buy all the items for Rs 9 in the following way:

- Buy the 1-st item for Rs 2 without using any tickets.
- Buy the 2-nd item for Rs 3 using 2 tickets.
- Buy the 3-rd item for Rs 4 using 1 ticket.

It can be shown that there is no way to buy all the items for less than Rs 9.

Problem B. Construct Binary Tree from Preorder and Inorder Traversal

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

Given two integer arrays a and b where a is the preorder traversal of a binary tree and b is the inorder traversal of the same tree, construct and return the binary tree.

Allowed Language(s): C, C++

Input

The first line of input contains n ($1 \leq n \leq 3000$) — the size of arrays a and b.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 3000$). It is guaranteed that the values of a will be unique.

The third line of input contains n space separated integers $b_1, b_2, \ldots, b_n$ ($1 \leq b_i \leq 3000$). It is guaranteed that the values of b will be unique.

Each value of a also appears in b.

Output

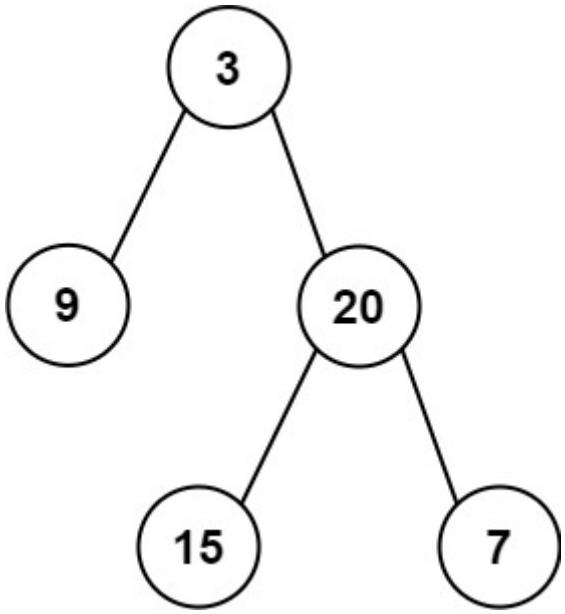
Print one line containing the elements of the array representing the binary tree. Follow the same format as specified in the **Directions for Binary Tree Problems**.

Examples

standard input	standard output
5 3 9 20 15 7 9 3 15 20 7	3 9 20 -1 -1 15 7

Explanation

The following figure illustrates the binary tree given in the example:



Problem C. Tatakae

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

Eren has entered the World of Paths. The World of Paths can be represented as a binary tree. Every node has a non-negative integer value associated with it. Eren is currently at the root node. For reasons that we cannot reveal without giving spoilers, Eren needs to reach a freedom node in the World of Paths. A freedom node is a node that satisfies all of the following conditions:

- It must be a leaf node.
- The sum of all the values on the path from the root node to the freedom node (both inclusive) must be exactly equal to a given integer, x.

Help Eren by finding the total number of freedom nodes in the World of Paths.

Note:

We define the path from node A to node B as a sequence of nodes starting with node A and ending with node B such that for every consecutive pair of nodes in the sequence, the second node is a left or right child of the first node.

The time complexity of your solution should be $O(m)$, where m is the number of nodes in the World of Paths. Solution of higher time complexity will not be considered.

Allowed Language(s): C, C++

Input

The first line of input contains two integers, n ($1 \leq n \leq 10^6$) and x ($0 \leq x \leq 10^9$) — the size of the array representing the binary tree and the sum required for freedom nodes.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($-1 \leq a_i \leq 10^9$).

Output

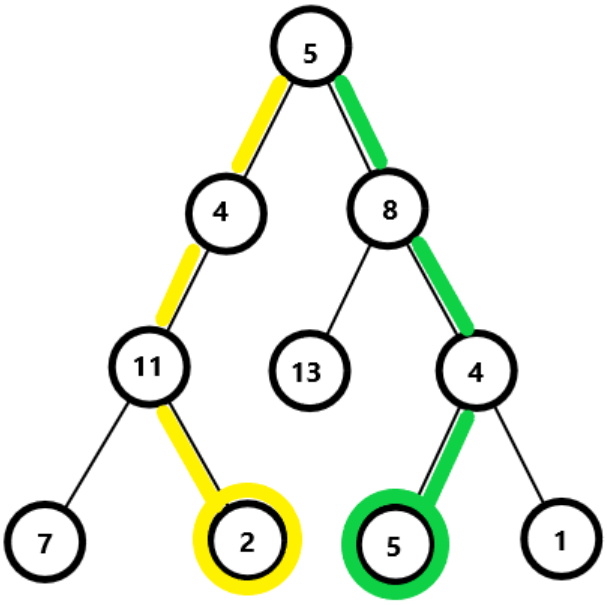
Print one line containing the number of freedom nodes.

Examples

standard input	standard output
15 22 5 4 8 11 -1 13 4 7 2 -1 -1 -1 -1 5 1	2

Explanation

The following figure illustrates the binary tree given in the example, with the freedom nodes and their paths highlighted. There are 2 freedom nodes in this tree.


5
4
8
11
13
4
7
2
5
1

Problem D. BST to GST

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

Given a Binary Search Tree (BST), convert it to a Greater Sum Tree (GST) such that every key of the original BST is changed to the original key plus the sum of all keys greater than the original key in the BST.

Note:

A binary search tree is a tree that satisfies the following constraints:

1. The left subtree of a node contains only nodes with keys less than the node's key.
2. The right subtree of a node contains only nodes with keys greater than the node's key.
3. Both the left and right subtrees must also be binary search trees.

Note that it follows from 1 and 2 in this case that no two nodes in a BST can have the same value.

The time complexity of your solution should be O(m), where m is the number of nodes in the BST. Solution with higher time complexity will not be considered.

Allowed Language(s): C, C++

Input

The first line of input contains n ($1 \leq n \leq 10^4$) — the size of the array representing the BST.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($-1 \leq a_i \leq 10^5$).

Output

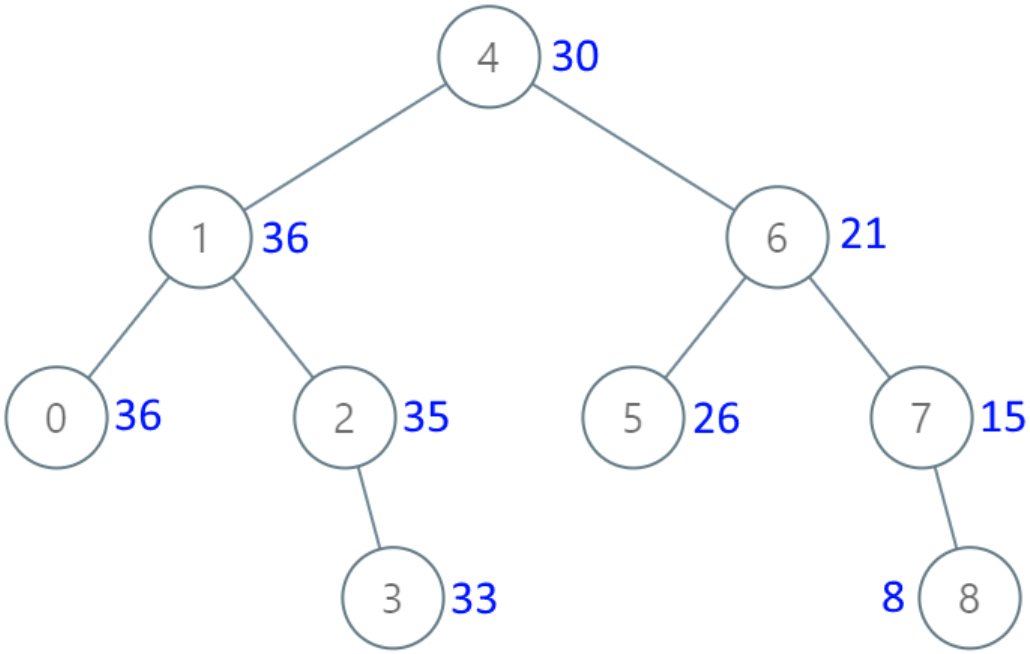
Print one line containing the elements of the array representing the GST. Follow the same format as specified in the **Directions for Binary Tree Problems**.

Example

standard input	standard output
15 4 1 6 0 2 5 7 -1 -1 -1 3 -1 -1 -1 8	30 36 21 36 35 26 15 -1 -1 -1 33 -1 -1 -1 8

Explanation

The following figure illustrates the BST given in the example:



For key 0 in the BST, the following keys had greater value : $1, 2, 3, 4, 5, 6, 7, 8$. Hence, the corresponding key for it in the GST is : $0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36$.

For key 2 in the BST, the following keys had greater value : $3, 4, 5, 6, 7, 8$. Hence, the corresponding key for it in the GST is : $3 + 4 + 5 + 6 + 7 + 8 = 35$.

Problem E. kth Largest in BST

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

Given a binary search tree, and an integer k, return the k-th largest value of all the values of the nodes in the tree.

Assume 1-based indexing.

The time complexity of your solution should be $O(h+k)$, where h is the height of the BST. Solutions with higher time complexity will not be considered.

Allowed Language(s): C, C++

Input

The first line of input contains n ($1 \leq n \leq 10^6$) — the size of the array representing the BST.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($-1 \leq a_i \leq 10^9$).

The third line of input contains k ($1 \leq k \leq$ the total number of nodes in the BST).

Output

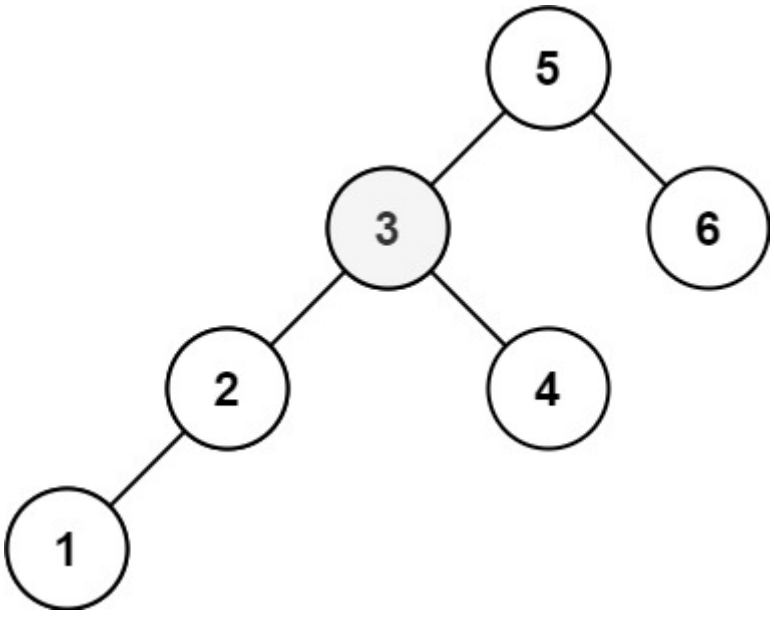
Print one line containing the k-th largest value in the BST.

Examples

standard input	standard output
8 5 3 6 2 4 -1 -1 1 4	3

Explanation

The following figure illustrates the BST given in the example:



Problem F. Fast Merge

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

As a part of your group project, you are given a huge amount of data to sort. You have k other students in your group and being the group leader you decide to divide the work among the k students. You divide the data into k parts (the parts need not be of equal sizes) and give each part to one of the students to sort. Every group member sorts his/her part and gives you a sorted array. Being the group leader, it is your duty to merge these sorted arrays into a single sorted array. Note that the size of the final sorted array is n such that $n = \Sigma_{i=1}^{k} m_i$

Languages allowed - C/C++

The time complexity of your solution should be $O(n * log(k))$. Any solution with a higher time complexity will not be considered.

Input

The first line consists of a single integer k — the number of students in your group.

Then k lines follow, the first integer of the i^{th} line is m_i ($1 \leq m \leq 10^4$), the size of the array given to the i^{th} student, m_i is followed by m_i space separated integers $a_{i1}, a_{i2}, \ldots, a_{im_i}$ ($1 \leq a_{ij} \leq 10^6$) — the elements of the array **after the i^{th} student sorts the array** given to him/her. (Look at examples for better understanding)

Note the additional constraint $\Sigma_{i=1}^{k} m_i \leq 10^5$ (i.e., the total number of elements is less than 10^5)

Output

Print n space separated integers — the elements of the merged sorted array. (Here n is the total size of the array and $n = \Sigma_{i=1}^{k} m_i$.

Examples

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

Note

In the first example, the first integer 4 denotes the number of group members (excluding you).

In the second line, the first integer 3, denotes the size of the first array and the 3 elements following it i.e., 1, 3 and 9 are the elements of the first array. Similarly the first integer of the second, third and fourth lines are the sizes of the arrays and they are followed by the elements of the array.

So, the 4 sorted arrays in the first example are - [1,3,9], [7,10], [3,4,7,9], [5,6,7] merging all of them gives [1,3,3,4,5,6,7,7,7,9,9,10]

Problem G. Customer Satisfaction

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

You are running a promotional event for Sugar Punch LLC. As part of your event, you hand out free chocolates to passersby. You have n boxes of chocolates. When handing out chocolates to a passerby, you select a box containing c chocolates and give $\lceil \frac{c}{2} \rceil$ chocolates to the passerby. The number of chocolates in the boxes need not be unique.

The satisfaction of a passerby is defined as the number of chocolates given to him/her. If you hand out chocolates to k passersby, find the maximum total satisfaction of the k passersby that can be achieved.

Note that you can give chocolates from the same box to multiple passersby and it may be possible that some passer(s)by does/do not get any chocolates.

Languages allowed - C, C++

The time complexity of your solution should be $O(k * logn)$. Any solution with a higher time complexity will not be considered.

Input

The first line of input contains two space separated integer, n ($1 \leq n \leq 10^6$) and k ($1 \leq k \leq 10^5$) — the number of boxes of chocolates and the number of passersby respectively. k can be less than, equal to or greater than n.

The second line of input contains n space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^6$) — a_i denotes the the number of chocolates initially in the i^{th} box. These integers need not be unique.

Output

Print a single integer - the maximum total satisfaction of the k passersby.

Examples

standard input	standard output
5 3 2 10 5 3 8	12
7 4 25 4 11 4 3 5 2	28

Note

In the first example, initially the chocolates in each box are - [2,10,5,3,8]

1. To the first person, we chose the box with 5 chocolates and give them 3 chocolates, the chocolates now are - [2,10,2,3,8]
2. To the second person, we chose the box with 10 chocolates and give them 5 chocolates, the chocolates now are - [2,5,2,3,8]
3. To the third person, we chose the box with 8 chocolates and give them 4 chocolates, the chocolates now are - [2,5,2,3,4]

The total satisfaction of the passersby is - 3 + 5 + 4 = 12

Note - There are different ways in which the chocolates can be distributed such that the total satisfaction is 12. There can also be a different order of choosing the boxes.

Problem H. Beautiful BST

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

You are given a binary search tree where the value of each node is a distinct positive integer. An integer is said to be **good** if it lies in the range $[l, r]$ (both l and r inclusive). A BST is said to be **beautiful** if it consists of only **good** integers.

Your task is to convert the given BST into a **Beautiful BST** by deleting the least number of nodes and preserving the relative order of the elements, i.e., any node's descendant in the original BST should remain a descendant in the new **Beautiful BST**.

Print the postorder traversal of the **Beautiful BST**.

Languages allowed - C, C++

The time complexity of your solution should be $O(m)$ where m is the number of nodes in the BST. Any solution with a higher time complexity will not be considered.

Input

The first line of the input contains three integers, n ($1 \leq n \leq 10^6$) l and r ($0 \leq l \leq r \leq 10^6$) — the size of the array representing the BST, the lower limit of the range and the upper limit of the range of good integers respectively.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ ($-1 \leq a_i \leq 10^9$) — the elements of the array representing the BST

It is guaranteed that at least one node in the BST will be a good integer.

Output

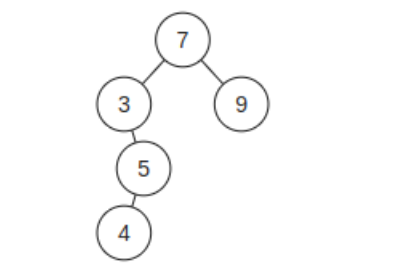
Print the post order traversal of the beautiful BST.

Examples

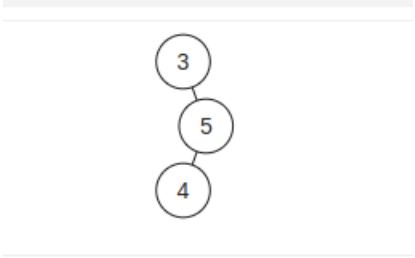
standard input	standard output
10 3 6 7 3 9 -1 5 -1 -1 -1 -1 4	4 5 3
7 6 11 9 6 12 4 7 11 15	7 6 11 9
7 2 10 3 2 6 -1 -1 5 9	2 5 9 6 3

Note

In the first example, the given BST looks like -

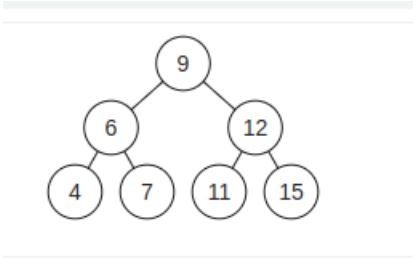


The given range is [3, 6], deleting all nodes outside the range, gives the following Beautiful BST

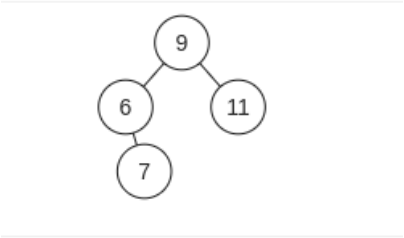


The post order traversal of this will be 4 5 3

In the second example, the given BST is -

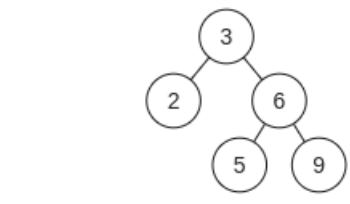


Removing nodes outside the range [6, 11] gives us



The postorder traversal of the above BST will be 7 6 11 9

In the third example, the BST is



The above BST is already beautiful, so the postorder traversal will be 2 5 9 6 3

Problem I. Ancestor

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

You are the captain of the spaceship Zenith. In one of your adventures, you come across the planet Zorath. The inhabitants of Zorath are called Zorathi. The Zorathi are peculiar creatures, in the sense that they are genderless and reproduce asexually. A single Zorathi can give birth to at most two children. In other words, **each Zorathi will have exactly one parent and can have up to two children.** Note that every Zorathi need not reproduce.

You come across two Zorathites (or Zorathi) who are related to each other and you want to find out how. To do so you decide to find the most recent ancestor they both have in common. You are given their family tree in the form of a **binary tree**. Each member in the family is represented as a number in the family tree and no two members have the same number. Find the most recent ancestor of two Zorathites.

Note - Every member of the family is an ancestor to himself/herself.

Languages allowed - C, C++

The time complexity of your solution should be $O(m)$ where m is the number of nodes in the binary tree. Any solution with a higher time complexity will not be considered.

Input

The first line of the input contains three integers, n $(1 \leq n \leq 10^6)$ x and y $(0 \leq x, y \leq n)$ — the size of the array representing the binary tree, the number associated with the first Zorathite and the number associated with the second Zorathite respectively. It is guaranteed that x and y will be distinct.

The second line of input contains n space separated integers $a_1, a_2, \ldots, a_n$ $(-1 \leq a_i \leq 10^9)$ — the elements of the array representing the binary tree.

Output

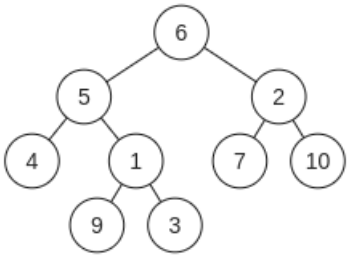
Print a single integer - the number associated with the most recent ancestor of the two Zorathites.

Examples

standard input	standard output
11 3 4 6 5 2 4 1 7 10 -1 -1 9 3	5
12 12 2 6 5 2 4 1 7 10 -1 -1 9 3 12	2

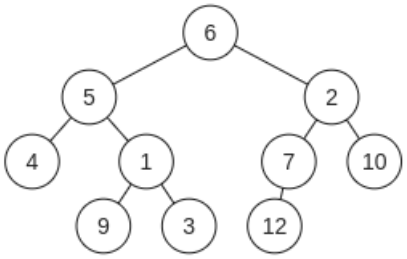
Note

In the first example, the family tree is



From the above tree, it is clear that the most recent common ancestor of 4 and 3 is 5.

In the second example, the family tree is



Since 2 is an ancestor to 12 and also to itself, 2 is the most recent common ancestor of 2 and 12.

Problem J. Combination Sum

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

You are given two arrays A and B consisting of n elements each. You construct another array C (initially empty) by performing the following operations (assume 1-based indexing) —

1. You pick a pair of indices (i, j) where $1 \leq i, j \leq n$
2. Pick the i^{th} element from A and the j^{th} element from B, add them and insert it to the end C. In other words, you insert $A_i + B_j$ to the end of C
3. Note that you cannot pick the same pair twice.

You do the above operation for all pairs of i and j and get the complete array C. Find the first k largest elements (not necessarily unique) of C.

Languages allowed - C, C++

The time complexity of your solution should be $O(n * log(n))$. Any solution with a higher time complexity will not be considered.

Input

The first line of input contains n, k $(1 \leq k \leq n \leq 10^6)$

The second line of input contains n space-separated integers $A_1, A_2, \ldots, A_n$ $(1 \leq A_i \leq 10^6)$ — A_i denotes the i^{th} element of A. The elements of A and the elements of B need not be unique.

The third line of input contains n space-separated integers $B_1, B_2, \ldots, B_n$ $(1 \leq B_i \leq 10^6)$ — B_i denotes the i^{th} element of B

Output

Print k space separated integers — the k largest elements of the array C.

Example

standard input	standard output
2 2 4 3 1 5	9 8
4 3 2 5 1 3 4 5 3 7	12 10 10

Note

In the first example, C after sorting in descending order will be [9, 8, 5, 4]
The two largest elements are 9, 8

In the second example, C after sorting in descending order is [12, 10, 10, 9, 9, 8, 8, 8, 7, 7, 6, 6, 6, 5, 5, 4]
The 3 largest elements are 12, 10, 10