**Split the array elements into strictly increasing and decreasing sequence**

Given an array of **N** elements. The task is to split the elements into two arrays say a1[] and a2[] such that one contains **strictly** **increasing** elements and the other contains **strictly decreasing** elements and **a1.size() + a2.size() = a.size()**. If it is not possible to do so, print **-1** or else print both the arrays.

**Note**: There can be multiple answers and the order of elements needs not to be the same in the child arrays.

**Examples:**

***Input:*** *a[] = {7, 2, 7, 3, 3, 1, 4}*

***Output:*** *a1[] = {1, 2, 3, 4, 7} , a2[] = {7, 3}*

***Input:*** *a[] = {1, 2, 2, 1, 1}*

***Output:*** *-1*

*It is not possile*

**Approach**: The following steps are followed to solve the above problem:

* **Initialize two vectors v1 and v2 which stores increasing and decreasing numbers.**
* **Use hashing to know the occurrence of the number in the array.**
* **If the number appears to come for the first time, then store it in v1.**
* **If the number appears to come for the second time, then store it in v2.**
* **If the number appears for more than 2 times, then it is not possible to store to create a strictly increasing and strictly decreasing array.**
* **At last, sort the first vector in increasing order and the second vector in decreasing order and print them.**

Below is the implementation of the above approach:

// Java program to implement

// the above approach

import java.util.\*;

class GFG

{

// Function to print both the arrays

static void PrintBothArrays(int a[], int n)

{

// Store both arrays

Vector<Integer> v1 = new Vector<Integer>(),

v2 = new Vector<Integer>();

// Used for hashing

HashMap<Integer, Integer> mpp = new HashMap<>();

// Iterate for every element

for (int i = 0; i < n; i++)

{

// Increase the count

mpp.put(a[i],(mpp.get(a[i]) == null?0:mpp.get(a[i]))+1);

// If first occurrence

if (mpp.get(a[i]) == 1)

v1.add(a[i]);

// If second occurrence

else if (mpp.get(a[i]) == 2)

v2.add(a[i]);

// If occurs more than 2 times

else

{

System.out.println( "Not possible");

return;

}

}

// Sort in increasing order

Collections.sort(v1);

// Print the increasing array

System.out.println("Strictly increasing array is:");

for (int i = 0; i < v1.size(); i++)

System.out.print(v1.get(i) + " ");

// Sort

Collections.sort(v2);

Collections.reverse(v2);

// Print the decreasing array

System.out.println("\nStrictly decreasing array is:");

for (int i = 0; i < v2.size(); i++)

System.out.print(v2.get(i) + " ");

}

// Driver code

public static void main(String args[])

{

int a[] = { 7, 2, 7, 3, 3, 1, 4 };

int n = a.length;

PrintBothArrays(a, n);

}

}

// This code is contributed by Arnab Kundu

**Output:**

Strictly increasing array is:

1 2 3 4 7

Strictly decreasing array is:

7 3