Kotlin Heroes: Practice 9

A. Spy Detected!

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array a consisting of n ($n \geq 3$) positive integers. It is known that in this array, all the numbers except one are the same (for example, in the array $[4, 11, 4, 4]$ all numbers except one are equal to 4).

Print the index of the element that does not equal others. The numbers in the array are numbered from one.

Input
The first line contains a single integer t ($1 \leq t \leq 100$). Then t test cases follow.

The first line of each test case contains a single integer n ($3 \leq n \leq 100$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 100$).

It is guaranteed that all the numbers except one in the a array are the same.

Output
For each test case, output a single integer — the index of the element that is not equal to others.

Example

input
4 4 11 13 11 11 5 1 4 4 4 4 10 3 3 3 3 10 3 3 3 3 3 3 20 20 10
output
2 1 5 3

B. Repeating Cipher

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp loves ciphers. He has invented his own cipher called *repeating*.

Repeating cipher is used for strings. To encrypt the string $s = s_1 s_2 \ldots s_m$ ($1 \leq m \leq 10$), Polycarp uses the following algorithm:

- he writes down s_1 ones,
- he writes down s_2 twice,
- he writes down s_3 three times,
- ...
- he writes down s_m m times.

For example, if s="bab" the process is: "b" $\rightarrow$ "baa" $\rightarrow$ "baabbb". So the encrypted s="bab" is "baabbb".

Given string t — the result of encryption of some string s. Your task is to decrypt it, i. e. find the string s.

Input
The first line contains integer n ($1 \leq n \leq 55$) — the length of the encrypted string. The second line of the input contains t — the result of encryption of some string s. It contains only lowercase Latin letters. The length of t is exactly n.

It is guaranteed that the answer to the test exists.

Output
Print such string s that after encryption it equals t.

Examples

input
6 baabbb
output
bab

input
10 oooppppssss
output
oops

input
1 z
output
z

C. Teams Forming

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n students in a university. The number of students is even. The i-th student has programming skill equal to a_i.

The coach wants to form $\frac{n}{2}$ teams. Each team should consist of exactly two students, and each student should belong to exactly one team. Two students can form a team only if their skills are equal (otherwise they cannot understand each other and cannot form a team).

Students can solve problems to increase their skill. One solved problem increases the skill by one.

The coach wants to know the minimum total number of problems students should solve to form exactly $\frac{n}{2}$ teams (i.e. each pair of students should form a team). Your task is to find this number.

Input
The first line of the input contains one integer n ($2 \le n \le 100$) — the number of students. It is guaranteed that n is even.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100$), where a_i is the skill of the i-th student.

Output
Print one number — the minimum total number of problems students should solve to form exactly $\frac{n}{2}$ teams.

Examples

input
6 5 10 2 3 14 5
output
5

input
2 1 100
output
99

Note
In the first example the optimal teams will be: $(3, 4)$, $(1, 6)$ and $(2, 5)$, where numbers in brackets are indices of students. Then, to form the first team the third student should solve 1 problem, to form the second team nobody needs to solve problems and to form the third team the second student should solve 4 problems so the answer is $1 + 4 = 5$.

In the second example the first student should solve 99 problems to form a team with the second one.

D. Two Shuffled Sequences

time limit per test: 3.0 s
memory limit per test: 256 megabytes

input: standard input
output: standard output

Two integer sequences existed initially — one of them was **strictly** increasing, and the other one — **strictly** decreasing.

Strictly increasing sequence is a sequence of integers $[x_1 < x_2 < \cdots < x_k]$. And strictly decreasing sequence is a sequence of integers $[y_1 > y_2 > \cdots > y_l]$. Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.

They were merged into one sequence a. After that sequence a got shuffled. For example, some of the possible resulting sequences a for an increasing sequence $[1, 3, 4]$ and a decreasing sequence $[10, 4, 2]$ are sequences $[1, 2, 3, 4, 4, 10]$ or $[4, 2, 1, 10, 4, 3]$.

This shuffled sequence a is given in the input.

Your task is to find **any** two suitable initial sequences. One of them should be **strictly** increasing and the other one — **strictly** decreasing. Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.

If there is a contradiction in the input and it is impossible to split the given sequence a to increasing and decreasing sequences, print "NO".

Input

The first line of the input contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of elements in a.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 2 \cdot 10^5$), where a_i is the i-th element of a.

Output

If there is a contradiction in the input and it is impossible to split the given sequence a to increasing and decreasing sequences, print "NO" in the first line.

Otherwise print "YES" in the first line and **any** two suitable sequences. Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.

In the second line print n_i — the number of elements in the **strictly increasing** sequence. n_i can be zero, in this case the increasing sequence is empty.

In the third line print n_i integers $inc_1, inc_2, \ldots, inc_{n_i}$ in the **increasing** order of its values ($inc_1 < inc_2 < \cdots < inc_{n_i}$) — the **strictly increasing** sequence itself. You can keep this line empty if $n_i = 0$ (or just print the empty line).

In the fourth line print n_d — the number of elements in the **strictly decreasing** sequence. n_d can be zero, in this case the decreasing sequence is empty.

In the fifth line print n_d integers $dec_1, dec_2, \ldots, dec_{n_d}$ in the **decreasing** order of its values ($dec_1 > dec_2 > \cdots > dec_{n_d}$) — the **strictly decreasing** sequence itself. You can keep this line empty if $n_d = 0$ (or just print the empty line).

$n_i + n_d$ should be equal to n and the union of printed sequences should be a permutation of the given sequence (in case of "YES" answer).

Examples

input
7 7 2 7 3 3 1 4
output
YES 2 3 7 5 7 4 3 2 1

input
5 4 3 1 5 3
output
YES 1 3 4 5 4 3 1

input
5 1 1 2 1 2
output
NO

input

5 0 1 2 3 4
output
YES 0 5 4 3 2 1 0

E. Powers Of Two

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

A positive integer x is called a *power of two* if it can be represented as $x = 2^y$, where y is a non-negative integer. So, the *powers of two* are $1, 2, 4, 8, 16, \ldots$.

You are given two positive integers n and k. Your task is to represent n as the **sum** of **exactly** k *powers of two*.

Input

The only line of the input contains two integers n and k ($1 \le n \le 10^9$, $1 \le k \le 2 \cdot 10^5$).

Output

If it is impossible to represent n as the sum of k powers of two, print `NO`.

Otherwise, print YES, and then print k positive integers $b_1, b_2, \ldots, b_k$ such that each of b_i is a power of two, and $\sum_{i=1}^{k} b_i = n$. If there are multiple answers, you may print any of them.

Examples

input
9 4
output
YES 1 2 2 4

input
8 1
output
YES 8

input
5 1
output
NO

input
3 7
output
NO

F. Boxers

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n boxers, the weight of the i-th boxer is a_i. Each of them can change the weight by no more than 1 before the competition (the weight cannot become equal to zero, that is, it must remain positive). Weight is always an integer number.

It is necessary to choose the largest boxing team in terms of the number of people, that all the boxers' weights in the team are different (i.e. unique).

Write a program that for given current values a_i will find the maximum possible number of boxers in a team.

It is possible that after some change the weight of some boxer is 150001 (but no more).

Input
The first line contains an integer n ($1 \le n \le 150000$) — the number of boxers. The next line contains n integers $a_1, a_2, \ldots, a_n$, where a_i ($1 \le a_i \le 150000$) is the weight of the i-th boxer.

Output
Print a single integer — the maximum possible number of people in a team.

Examples

input
4 3 2 4 1
output
4

input
6 1 1 1 4 4 4
output
5

Note
In the first example, boxers should not change their weights — you can just make a team out of all of them.

In the second example, one boxer with a weight of 1 can be increased by one (get the weight of 2), one boxer with a weight of 4 can be reduced by one, and the other can be increased by one (resulting the boxers with a weight of 3 and 5, respectively). Thus, you can get a team consisting of boxers with weights of $5, 4, 3, 2, 1$.

G. Median String

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two strings s and t, both consisting of exactly k lowercase Latin letters, s is lexicographically less than t.

Let's consider list of all strings consisting of exactly k lowercase Latin letters, lexicographically not less than s and not greater than t (including s and t) in lexicographical order. For example, for $k = 2$, $s =$"az" and $t =$"bf" the list will be ["az", "ba", "bb", "bc", "bd", "be", "bf"].

Your task is to print the median (the middle element) of this list. For the example above this will be "bc".

It is guaranteed that there is an odd number of strings lexicographically not less than s and not greater than t.

Input
The first line of the input contains one integer k ($1 \le k \le 2 \cdot 10^5$) — the length of strings.

The second line of the input contains one string s consisting of exactly k lowercase Latin letters.

The third line of the input contains one string t consisting of exactly k lowercase Latin letters.

It is guaranteed that s is lexicographically less than t.

It is guaranteed that there is an odd number of strings lexicographically not less than s and not greater than t.

Output
Print one string consisting exactly of k lowercase Latin letters — the median (the middle element) of list of strings of length k lexicographically not less than s and not greater than t.

Examples

input
2 az bf
output
bc

input
5 afogk asdji
output
alvuw

input
6 nijfvj tvqhwp
output
qoztvz

H. Two Merged Sequences

time limit per test: 3.0 s
memory limit per test: 256 megabytes
input: standard input
output: standard output

Two integer sequences existed initially, one of them was **strictly** increasing, and another one — **strictly** decreasing.

Strictly increasing sequence is a sequence of integers $[x_1 < x_2 < \cdots < x_k]$. And strictly decreasing sequence is a sequence of integers $[y_1 > y_2 > \cdots > y_l]$. Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.

Elements of increasing sequence were inserted between elements of the decreasing one (and, possibly, before its first element and after its last element) **without changing the order**. For example, sequences $[1, 3, 4]$ and $[10, 4, 2]$ can produce the following resulting sequences: $[10, \mathbf{1}, \mathbf{3}, 4, 2, \mathbf{4}]$, $[\mathbf{1}, \mathbf{3}, \mathbf{4}, 10, 4, 2]$. The following sequence cannot be the result of these insertions: $[\mathbf{1}, 10, \mathbf{4}, 4, \mathbf{3}, 2]$ because the order of elements in the increasing sequence was changed.

Let the obtained sequence be a. This sequence a is given in the input. Your task is to find **any** two suitable initial sequences. One of them should be **strictly** increasing, and another one — **strictly** decreasing. **Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.**

If there is a contradiction in the input and it is impossible to split the given sequence a into one increasing sequence and one decreasing sequence, print "NO".

Input

The first line of the input contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of elements in a.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 2 \cdot 10^5$), where a_i is the i-th element of a.

Output

If there is a contradiction in the input and it is impossible to split the given sequence a into one increasing sequence and one decreasing sequence, print "NO" in the first line.

Otherwise print "YES" in the first line. In the second line, print a sequence of n integers $res_1, res_2, \ldots, res_n$, where res_i should be either 0 or 1 for each i from 1 to n. The i-th element of this sequence should be 0 if the i-th element of a belongs to the increasing sequence, and 1 otherwise. **Note that the empty sequence and the sequence consisting of one element can be considered as increasing or decreasing.**

Examples

input
9 5 1 3 6 8 2 9 0 10
output
YES 1 0 0 0 0 1 0 1 0

input
5 1 2 4 0 2
output
NO