

Maximize XOR sum

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 512 megabytes

Let us use $\oplus$ as the symbol for the operation of *bitwise "exclusive or"* for integers. In C++ and Java it is denoted by the character "`^`", in Pascal and Python — by the keyword "`xor`". For example, $9 \oplus 3 = 1001_2 \oplus 11_2 = 1010_2 = 10$.

You are given two integer arrays A and B of length n . Let's denote $X(A)$ as the result of calculating bitwise "exclusive or" for all elements of the array: $X(A) = A_1 \oplus A_2 \oplus \dots \oplus A_n$. Similarly, let's denote $X(B) = B_1 \oplus B_2 \oplus \dots \oplus B_n$.

For each i from 1 to n , it is allowed to swap elements A_i and B_i . You must find out which elements should be swapped in order for the sum $X(A) + X(B)$ to be maximum possible.

Input

The first line contains an integer n — the size of the arrays ($1 \leq n \leq 10^5$). The next line contains n integers A_i — elements of the array A ($0 \leq A_i \leq 10^{18}$). The next line contains the array B in the same format.

Output

The first line of output must contain the maximum possible sum $X(A) + X(B)$ and an integer k — the number of required swaps. The next line must contain k different integers from 1 to n — indices of the elements to be swapped.

Example

standard input	standard output
2	6 1
1 1	1
2 2	

Note

In the example after the swap the arrays are $A = [2, 1]$ and $B = [1, 2]$.

$X(A) = 2 \oplus 1 = 10_2 \oplus 1_2 = 11_2 = 3$, $X(B) = 3$, $X(A) + X(B) = 6$.