

# Hieroglyphs

A team of researchers is studying the similarities between sequences of hieroglyphs. They represent each hieroglyph with a non-negative integer. To perform their study, they use the following concepts about sequences.

For a fixed sequence  $A$ , a sequence  $S$  is called a **subsequence** of  $A$  if and only if  $S$  can be obtained by removing some elements (possibly none) from  $A$ .

The table below shows some examples of subsequences of a sequence  $A = [3, 2, 1, 2]$ .

Subsequence	How it can be obtained from $A$
$[3, 2, 1, 2]$	No elements are removed.
$[2, 1, 2]$	$[\cancel{3}, 2, 1, 2]$
$[3, 2, 2]$	$[3, 2, \cancel{1}, 2]$
$[3, 2]$	$[3, \cancel{2}, \cancel{1}, 2]$ or $[3, 2, \cancel{1}, \cancel{2}]$
$[3]$	$[3, \cancel{2}, \cancel{1}, \cancel{2}]$
$[\ ]$	$[\cancel{3}, \cancel{2}, \cancel{1}, \cancel{2}]$

On the other hand,  $[3, 3]$  or  $[1, 3]$  are not subsequences of  $A$ .

Consider two sequences of hieroglyphs,  $A$  and  $B$ . A sequence  $S$  is called a **common subsequence** of  $A$  and  $B$  if and only if  $S$  is a subsequence of both  $A$  and  $B$ . Moreover, we say that a sequence  $U$  is a **universal common subsequence** of  $A$  and  $B$  if and only if the following two conditions are met:

- $U$  is a common subsequence of  $A$  and  $B$ .
- Every common subsequence of  $A$  and  $B$  is also a subsequence of  $U$ .

It can be shown that any two sequences  $A$  and  $B$  have at most one universal common subsequence.

The researchers have found two sequences of hieroglyphs  $A$  and  $B$ . Sequence  $A$  consists of  $N$  hieroglyphs and sequence  $B$  consists of  $M$  hieroglyphs. Help the researchers compute a universal common subsequence of sequences  $A$  and  $B$ , or determine that such a sequence does not exist.

## Implementation details

You should implement the following procedure.

```
std::vector<int> ucs(std::vector<int> A, std::vector<int> B)
```

- $A$ : array of length  $N$  describing the first sequence.
- $B$ : array of length  $M$  describing the second sequence.
- If there exists a universal common subsequence of  $A$  and  $B$ , the procedure should return an array containing this sequence. Otherwise, the procedure should return  $[-1]$  (an array of length 1, whose only element is  $-1$ ).
- This procedure is called exactly once for each test case.

## Constraints

- $1 \leq N \leq 100\,000$
- $1 \leq M \leq 100\,000$
- $0 \leq A[i] \leq 200\,000$  for each  $i$  such that  $0 \leq i < N$
- $0 \leq B[j] \leq 200\,000$  for each  $j$  such that  $0 \leq j < M$

## Subtasks

Subtask	Score	Additional Constraints
1	3	$N = M$ ; each of $A$ and $B$ consists of $N$ <b>distinct</b> integers between 0 and $N - 1$ (inclusive)
2	15	For any integer $k$ , (the number of elements of $A$ equal to $k$ ) plus (the number of elements of $B$ equal to $k$ ) is at most 3.
3	10	$A[i] \leq 1$ for each $i$ such that $0 \leq i < N$ ; $B[j] \leq 1$ for each $j$ such that $0 \leq j < M$
4	16	There exists a universal common subsequence of $A$ and $B$ .
5	14	$N \leq 3000$ ; $M \leq 3000$
6	42	No additional constraints.

## Examples

### Example 1

Consider the following call.

```
ucs([0, 0, 1, 0, 1, 2], [2, 0, 1, 0, 2])
```

Here, the common subsequences of  $A$  and  $B$  are the following: [], [0], [1], [2], [0,0], [0,1], [0,2], [1,0], [1,2], [0,0,2], [0,1,0], [0,1,2], [1,0,2] and [0,1,0,2].

Since [0,1,0,2] is a common subsequence of  $A$  and  $B$ , and all common subsequences of  $A$  and  $B$  are subsequences of [0,1,0,2], the procedure should return [0,1,0,2].

### Example 2

Consider the following call.

```
ucs([0, 0, 2], [1, 1])
```

Here, the only common subsequence of  $A$  and  $B$  is the empty sequence []. It follows that the procedure should return an empty array [].

### Example 3

Consider the following call.

```
ucs([0, 1, 0], [1, 0, 1])
```

Here, the common subsequences of  $A$  and  $B$  are [], [0], [1], [0,1] and [1,0]. It can be shown that a universal common subsequence does not exist. Therefore, the procedure should return [-1].

## Sample Grader

Input format:

```
N M
A[0] A[1] ... A[N-1]
B[0] B[1] ... B[M-1]
```

Output format:

```
T
R[0] R[1] ... R[T-1]
```

Here,  $R$  is the array returned by `ucs` and  $T$  is its length.