You are given a **0-indexed** integer array nums containing **distinct** numbers, an integer start, and an integer goal. There is an integer x that is initially set to start, and you want to perform operations on x such that it is converted to goal. You can perform the following operation repeatedly on the number x:

If 0 <= x <= 1000, then for any index i in the array (0 <= i < nums.length), you can set x to any of the following:

* x + nums[i]
* x - nums[i]
* x ^ nums[i] (bitwise-XOR)

Note that you can use each nums[i] any number of times in any order. Operations that set x to be out of the range 0 <= x <= 1000 are valid, but no more operations can be done afterward.

Return *the****minimum****number of operations needed to convert*x = start*into*goal*, and*-1*if it is not possible*.

**Example 1:**

**Input:** nums = [1,3], start = 6, goal = 4

**Output:** 2

**Explanation:**

We can go from 6 → 7 → 4 with the following 2 operations.

- 6 ^ 1 = 7

- 7 ^ 3 = 4

**Example 2:**

**Input:** nums = [2,4,12], start = 2, goal = 12

**Output:** 2

**Explanation:**

We can go from 2 → 14 → 12 with the following 2 operations.

- 2 + 12 = 14

- 14 - 2 = 12

**Example 3:**

**Input:** nums = [3,5,7], start = 0, goal = -4

**Output:** 2

**Explanation:**

We can go from 0 → 3 → -4 with the following 2 operations.

- 0 + 3 = 3

- 3 - 7 = -4

Note that the last operation sets x out of the range 0 <= x <= 1000, which is valid.

**Example 4:**

**Input:** nums = [2,8,16], start = 0, goal = 1

**Output:** -1

**Explanation:**

There is no way to convert 0 into 1.

**Example 5:**

**Input:** nums = [1], start = 0, goal = 3

**Output:** 3

**Explanation:**

We can go from 0 → 1 → 2 → 3 with the following 3 operations.

- 0 + 1 = 1

- 1 + 1 = 2

- 2 + 1 = 3

**Constraints:**

* 1 <= nums.length <= 1000
* -109 <= nums[i], goal <= 109
* 0 <= start <= 1000
* start != goal
* All the integers in nums are distinct.