Extra DSA questions

Array manipulation

1) Find sum indices:

-> User will give give an Array of integers ***nums*** of n size and an Integer ***target***. You have to find 2 indices such that sum of those elements is equal to target. (Considering 2 indices to be x and y, nums[x]+nums[y]=target)

Example 1:

Input: nums = [2,7,11,15], target = 9

Output: [0,1]

Explanation: Because nums[0] + nums[1] == 9, we return [0,1].

Example 2:

Input: nums = [3,2,4], target = 6

Output: [1,2]

Example 3:

Input: nums = [3,3], target = 6

Output: [0,1]

2) Search insert position :

-> You are given sorted array of integers and a target value. Return the index if the target is found. If not, return the index where it would be if it were inserted in order.
(Improvement : Use Binary search to find the index)

Example 1:

Input: nums = [1,3,5,6], target = 5

Output: 2

Example 2:

Input: nums = [1,3,5,6], target = 2

Output: 1

Example 3:

Input: nums = [1,3,5,6], target = 7

Output: 4

3) Single number :

-> Given a **non-empty** array of integers nums, every element appears *twice* except for one. Find that single one.
(Improvement : Try doing it in single looping)

Constraints:

array values range : -10000 to 10000

Example 1:

Input: nums = [2,2,1]

Output: 1

Example 2:

Input: nums = [4,1,2,1,2]

Output: 4

Example 3:

Input: nums = [1]

Output: 1

4) Intersection of arrays :

-> Given two integer arrays nums1 and nums2, return *an array of their intersection*. Each element in the result must appear as many times as it shows in both arrays. Order of answer doesn't matter.

Constraints:

Array values range : 0 to 1000

Example 1:

Input: nums1 = [1,2,2,1], nums2 = [2,2]

Output: [2,2]

Example 2:

Input: nums1 = [4,9,5], nums2 = [9,4,9,8,4]

Output: [4,9]

[9,4] is also accepted.

5) Maximize sum of pairs :

You are given array of even length n, named ***nums***. You have to pair up every element (you can ignore the order). Value of each pair would be minimum of both (if pair = (5,6), value =5). We have to maximize sum of value of all pairs and return it as answer.

Constraints:

Length of array should be even (n%2 == 0).

Example 1:

Input: nums = [1,4,3,2]

Output: 4

Explanation: All possible pairings (ignoring the ordering of elements) are:

1. (1, 4), (2, 3) -> min(1, 4) + min(2, 3) = 1 + 2 = 3
2. (1, 3), (2, 4) -> min(1, 3) + min(2, 4) = 1 + 2 = 3
3. (1, 2), (3, 4) -> min(1, 2) + min(3, 4) = 1 + 3 = 4

So the maximum possible sum is 4.

Example 2:

Input: nums = [6,2,6,5,1,2]

Output: 9

Explanation: The optimal pairing is (2, 1), (2, 5), (6, 6). min(2,1) + min(2, 5) + min(6, 6) = 1 + 2 + 6 = 9.

Improvement :

6) Manage change :

-> There's a lemonade stand, where each one cost for 5 Rs. Customers who buy it, pay with 5 Rs., 10 Rs. and 15 Rs. You are given an array queue of people paying for lemonade.

You have to tell if we can manage to return change to every customer without breaking queue.

Constraints:

Array values – anyone of 5, 10 and 15.

Example 1:

Input: bills = [5,5,5,10,20]

Output: true

Explanation:

From the first 3 customers, we collect three $5 bills in order.

From the fourth customer, we collect a $10 bill and give back a $5.

From the fifth customer, we give a $10 bill and a $5 bill.

Since all customers got correct change, we output true.

Example 2:

Input: bills = [5,5,10,10,20]

Output: false

Explanation:

From the first two customers in order, we collect two $5 bills.

For the next two customers in order, we collect a $10 bill and give back a $5 bill.

For the last customer, we can not give the change of $15 back because we only have two $10 bills.

Since not every customer received the correct change, the answer is false.

7) Merge two sorted arrays :

-> You are given 2 arrays, nums1 and nums2 with length of m and n, sorted in ascending order. You have to create new array of (m + n) length, that has all elements from both elements, in sorted order.

Improvement : Solve problem with maximum (m+n) iterations.

Example 1:

Input: nums1 = [1,2,3,0,0,0], m = 3, nums2 = [2,5,6], n = 3

Output: [1,2,2,3,5,6]

Explanation: The arrays we are merging are [1,2,3] and [2,5,6].

The result of the merge is [1,2,2,3,5,6] with the underlined elements coming from nums1.

Example 2:

Input: nums1 = [1], m = 1, nums2 = [], n = 0

Output: [1]

Explanation: The arrays we are merging are [1] and [].

The result of the merge is [1].

Example 3:

Input: nums1 = [0], m = 0, nums2 = [1], n = 1

Output: [1]

Explanation: The arrays we are merging are [] and [1].

The result of the merge is [1].

Note that because m = 0, there are no elements in nums1.

The 0 is only there to ensure the merge result can fit in nums1.

8) Move chips to same spot :

-> Chips are piled up in different locations. You are given array ***position*** containing location of each chip.

You have to pile-up all chips on one locations by performing the following operations :

i) move a chip 2 places forward or backward with cost = 0.

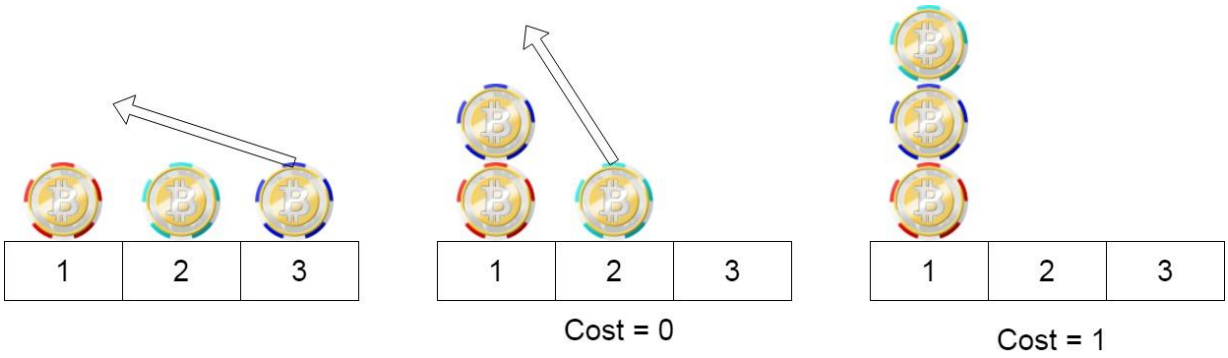
ii) move a chip 1 place forward or backward with cost = 1.

You cannot move out of array at any number of operation.

Return cost to pile-up all chips on one location.

improvement : Do it without count array and in O(n) time.

Example 1:



Input: position = [1,2,3]

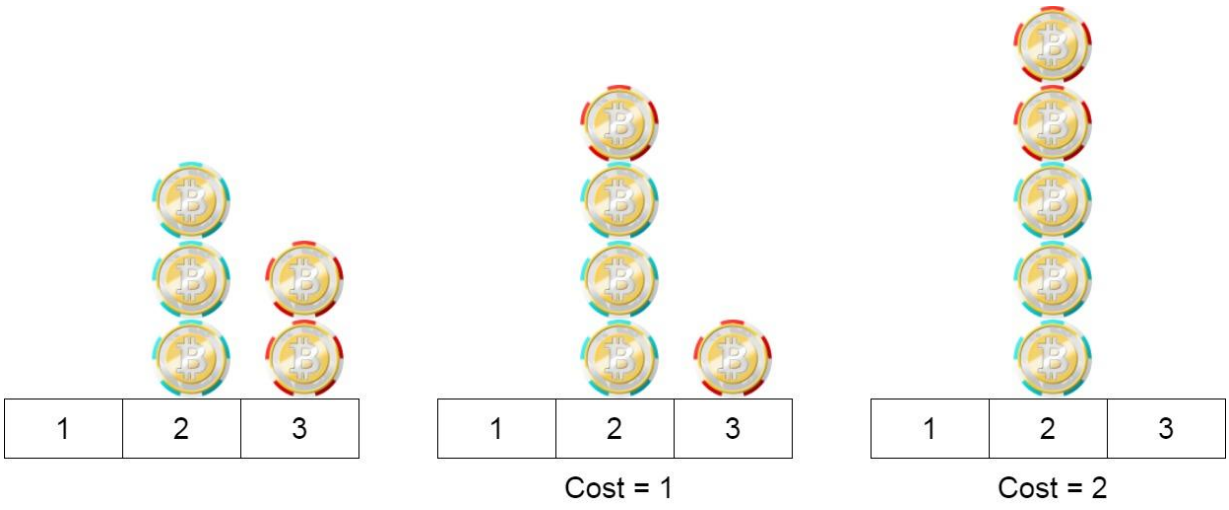
Output: 1

Explanation: First step: Move the chip at position 3 to position 1 with cost = 0.

Second step: Move the chip at position 2 to position 1 with cost = 1.

Total cost is 1.

Example 2:



Input: position = [2,2,2,3,3]

Output: 2

Explanation: We can move the two chips at position 3 to position 2. Each move has cost = 1. The total cost = 2.

Example 3:

Input: position = [1,1000000000]

Output: 1

9) De-XOR array :

-> An array of length n is XORed by doing performing following for first n-1 elements :

xored_array[i] = original_array[i] ^ original_array[i+1] (here, A^B is A XOR B)

You will be given XORed array ***xored_array*** and first element of original_array. You have to create original_array from it.

Brief about xor (^) :

XOR gate follows given table for each bit

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

I) For 2 values A and B, if A^B = C,

then, A^C = B and B^C = A.

II) A^B = B ^A

III) A^A = 0

IV) (A^B) ^ C = A^ (B^ C)

Example 1:

Input: xored_array = [1,2,3], first = 1

Output: [1,0,2,1]

Explanation: If original_array = [1,0,2,1], then first = 1 and encoded = [1 XOR 0, 0 XOR 2, 2 XOR 1] = [1,2,3]

Example 2:

Input: xored_array = [6,2,7,3], first = 4

Output: [4,2,0,7,4]

10) Total rabbits in forest :

-> A forest is having some number of rabbits. A survey is done by asking n rabbits “How many rabbits have same color as you?”. Later on, That survey is converted to array, Given as ***answers***. We have to find minimum numbers of rabbits that could be in forest from given array.

Example 1:

Input: answers = [1,1,2]

Output: 5

Explanation:

The two rabbits that answered "1" could both be the same color, say red.

The rabbit that answered "2" can't be red or the answers would be inconsistent.

Say the rabbit that answered "2" was blue.

Then there should be 2 other blue rabbits in the forest that

didn't answer into the array.

The smallest possible number of rabbits in the forest is therefore 5: 3 that answered plus 2 that didn't.

Example 2:

Input: answers = [10,10,10]

Output: 11