ARRAYLIST QUESTIONS

Question 1 :

Monotonic ArrayList (EASY)

An Arraylist is monotonic if it is either monotone increasing or monotone decreasing.

An Arraylist nums is monotone increasing if for all i <= j, nums.get(i) <= nums.get(j). An Arraylist nums is monotone decreasing if for all i <= j, nums.get(i) >= nums.get(j).

Given an integer Arraylist nums, return true if the given list is monotonic, or false otherwise.

Sample Input 1 : nums = [1,2,2,3]
Sample Output 1 : true

Sample Input 2 : nums = [6,5,4,4]
Sample Output 2 : true

Sample Input 3 : nums = [1,3,2]
Sample Output 3 : false

Constraints :

- 1 <= nums.size() <= 105
- -105 <= nums.get(i) <= 105

Question 2 :

Lonely Numbers in ArrayList (MEDIUM)

You are given an integer arraylist nums. A number x is lonely when it appears only once, and no adjacent numbers (i.e. x + 1 and x - 1) appear in the arraylist.

Return all lonely numbers in nums. You may return the answer in any order.

Sample Input 1 : nums = [10,6,5,8]
Sample Output 1 : [10,8]
Explanation :
- 10 is a lonely number since it appears exactly once and 9 and 11 does not appear in nums.
- 8 is a lonely number since it appears exactly once and 7 and 9 does not appear in nums.

- 5 is not a lonely number since 6 appears in nums and vice versa.
Hence, the lonely numbers in nums are [10, 8].
Note that [8, 10] may also be returned.

Sample Input 2 : nums = [1,3,5,3]
Sample Output 2 : [1,5]
Explanation :
- 1 is a lonely number since it appears exactly once and 0 and 2 does not appear in nums.
- 5 is a lonely number since it appears exactly once and 4 and 6 does not appear in nums.
- 3 is not a lonely number since it appears twice.
Hence, the lonely numbers in nums are [1, 5].
Note that [5, 1] may also be returned.

Constraints :

- 1 <= nums.size() <= 105
- 0 <= nums.get(i) <= 106

Question 3 :

Most Frequent Number following Key (EASY)

You are given an integer Arraylist nums. You are also given an integer key, which is present in nums.

For every unique integer target in nums, count the number of times target immediately follows an occurrence of key in nums. In other words, count the number of indices i such that:

0 <= i <= nums.size() - 2,
nums.get(i) == key and,
nums.get(i+1) == target.
Return the target with the maximum count.

(Assumption - that the target with maximum count is unique.)

Sample Input 1 :nums = [1,100,200,1,100], key = 1
Sample Output 1 : 100
Explanation :
For target = 100, there are 2 occurrences at indices 1 and 4 which follow an occurrence of key.
No other integers follow an occurrence of key, so we return 100.

Sample Input 2 : nums = [2,2,2,2,3], key = 2
Sample Output 2 : 2
Explanation :
For target = 2, there are 3 occurrences at indices 1, 2, and 3 which follow an occurrence of key.
For target = 3, there is only one occurrence at index 4 which follows an occurrence of key.
target = 2 has the maximum number of occurrences following an occurrence of key, so we return 2.

Constraints :

- 2 <= nums.size() <= 1000
- 1 <= nums.get(i) <= 1000
- Assume that the answer is unique.

Hints : Count the number of times each target value follows the key in the arraylist.
Choose the target with the maximum count and return it.

Question 4 :

Beautiful ArrayList (MEDIUM)
An Arraylist nums of size n is beautiful if:

nums is a permutation of the integers in the range [1, n].
For every 0 <= i < j < n, there is no index k with i < k < j where 2 * nums.get(k) == nums.get(i) + nums.get(j).
Given the integer n, return any beautiful arraylist nums of size n. There will be at least one valid answer for the given n.

Sample Input 1 : n = 4
Sample Output 1 : [2,1,4,3]

Sample Input 2 : n = 5
Sample Output 2 : [3,1,2,5,4]

Constraints :

- 1 <= n <= 1000