**525. Contiguous Array**

Given a binary array nums, return *the maximum length of a contiguous subarray with an equal number of*0*and*1.

**Input:** nums = [0,1]

**Output:** 2

**Explanation:** [0, 1] is the longest contiguous subarray with an equal number of 0 and 1.

Sol:

Instead of 0 mark it to -1…now find the longest subarray with maximum sum.

**523. Continuous Subarray Sum**

Given an integer array nums and an integer k, return true *if*nums*has a continuous subarray of size****at least two****whose elements sum up to a multiple of* k*, or*false*otherwise*.

An integer x is a multiple of k if there exists an integer n such that x = n \* k. 0 is **always** a multiple of k.

**Input:** nums = [23,2,4,6,7], k = 6

**Output:** true

**Explanation:** [2, 4] is a continuous subarray of size 2 whose elements sum up to 6.

Sol:

Put remainder as key in map and index as value.

**791. Custom Sort String**

You are given two strings order and s. All the characters of order are **unique** and were sorted in some custom order previously.

Permute the characters of s so that they match the order that order was sorted. More specifically, if a character x occurs before a character y in order, then x should occur before y in the permuted string.

Return *any permutation of*s*that satisfies this property*.

**Input:** order = "cba", s = "abcd"

**Output:** "cbad"

**Explanation:**

"a", "b", "c" appear in order, so the order of "a", "b", "c" should be "c", "b", and "a".

Since "d" does not appear in order, it can be at any position in the returned string. "dcba", "cdba", "cbda" are also valid outputs.

**287. Find the Duplicate Number**

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.

There is only **one repeated number** in nums, return *this repeated number*.

You must solve the problem **without** modifying the array nums and uses only constant extra space.

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

**Output:** 2

**287. Find the Duplicate Number**

Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive.

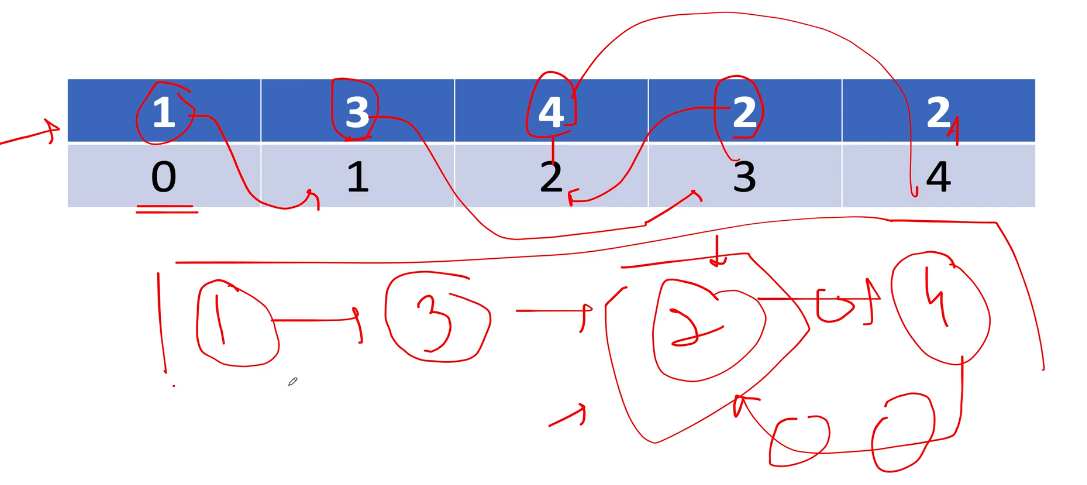
There is only **one repeated number** in nums, return *this repeated number*.

You must solve the problem **without** modifying the array nums and uses only constant extra space.

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

**Output:** 2

Sol:



We start from 1st index whose value is 1….create a node(1)

Now go to index 1 whose value is 3 …..create a node(3)

Now go to index 3 whose value is 2……create a node(2)

Now go to index 2 whose value is 4…..create a node(4)

Now go to index 4 whose value is 2……it forms a cycle.

And start node of the cycle is duplicate number.

**442. Find All Duplicates in an Array**

Given an integer array nums of length n where all the integers of nums are in the range [1, n] and each integer appears **once** or **twice**, return *an array of all the integers that appears****twice***.

You must write an algorithm that runs in O(n) time and uses only constant extra space.

**Input:** nums = [4,3,2,7,8,2,3,1]

**Output:** [2,3]

**1838. Frequency of the Most Frequent Element**

The **frequency** of an element is the number of times it occurs in an array.

You are given an integer array nums and an integer k. In one operation, you can choose an index of nums and increment the element at that index by 1.

Return *the****maximum possible frequency****of an element after performing****at most***k*operations*.

**Input:** nums = [1,2,4], k = 5

**Output:** 3

**Explanation:** Increment the first element three times and the second element two times to make nums = [4,4,4].

4 has a frequency of 3.

**491. Increasing Subsequences**

Given an integer array nums, return all the different possible increasing subsequences of the given array with **at least two elements**. You may return the answer in **any order**.

The given array may contain duplicates, and two equal integers should also be considered a special case of increasing sequence.

**Input:** nums = [4,6,7,7]

**Output:** [[4,6],[4,6,7],[4,6,7,7],[4,7],[4,7,7],[6,7],[6,7,7],[7,7]]

**334. Increasing Triplet Subsequence**

Given an integer array nums, return true*if there exists a triple of indices*(i, j, k)*such that*i < j < k*and*nums[i] < nums[j] < nums[k]. If no such indices exists, return false.

**Input:** nums = [1,2,3,4,5]

**Output:** true

**Explanation:** Any triplet where i < j < k is valid.

**55. Jump Game**

You are given an integer array nums. You are initially positioned at the array's **first index**, and each element in the array represents your maximum jump length at that position.

Return true*if you can reach the last index, or*false*otherwise*.

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

**Output:** true

**Explanation:** Jump 1 step from index 0 to 1, then 3 steps to the last index.

**45. Jump Game II**

Given an array of non-negative integers nums, you are initially positioned at the first index of the array.

Each element in the array represents your maximum jump length at that position.

Your goal is to reach the last index in the minimum number of jumps.

You can assume that you can always reach the last index.

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

**Output:** 2

**Explanation:** The minimum number of jumps to reach the last index is 2. Jump 1 step from index 0 to 1, then 3 steps to the last index.