UCS301 Data Structures

Lab Assignment 10

Hashing

1. Given an integer array, check if it contains any **duplicates** using hash set.

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

2. Given two arrays, find the **common elements** using a hash set.

   Input:
   A = [1, 2, 3, 4]
   B = [3, 4, 5, 6]
   Output:
   3, 4

3. Count the **frequency** of each number in an array using a hash map.

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

   Output:

   - 2 → 3 times
   - 3 → 2 times
   - 4 → 1 time

4. Find the **first non-repeating element** in an array using a hash map.

   Input:
   [4, 5, 1, 2, 0, 4]
   Output:
   5

5. Given a **linked list**, determine whether it contains a **loop (cycle)** using a hash set.
   A loop exists if some node's next pointer points to a **previous node** in the list.

   Input:
   1 → 2 → 3 → 4 → 2 (back to node 2)
   Output:
   true

6. You are given a **binary tree (not a binary search tree)**. Write a program to check **if the tree contains any duplicate node values** using hash set.
   Your task is to:
   1. Return true (or print "Duplicates Found") if there exists at least one duplicate value.
   2. Otherwise, return false (or print "No Duplicates").

Additional Questions

Q1. Given an integer array **arr[]**, find the element that appears most frequently. If multiple elements have the same highest frequency, return the largest among them.

Examples:

- ***Input :*** *arr[] = [1, 3, 2, 1, 4, 1]*
  Output : *1*
  Explanation: *1 appears three times in array which is maximum frequency.*
- ***Input :*** *arr[] = [10, 20, 10, 20, 30, 20, 20]*
  Output : *20 appears four times in array which is maximum frequency*
- ***Input:*** *arr[] = [1, 2, 2, 4, 1]*
  Output: *2*
  Explanation: *1 and 2 both appear two times, so return 2 as it's value is bigger.*

Q2. Given an array **arr[]** of n integers and a **target** value, check if there exists a pair whose sum equals the target.

Examples:

- **Input**: arr[] = [0, -1, 2, -3, 1], target = -2
  Output: true
  Explanation: There is a pair (1, -3) with the sum equal to given target, 1 + (-3) = -2.
- **Input**: arr[] = [1, -2, 1, 0, 5], target = 0
  Output: false
  Explanation: There is no pair with sum equals to given target.

Q3. Given two arrays, **a** and **b** of equal length. The task is to determine if the given arrays are equal or not. Two arrays are considered equal if:

- Both arrays contain the same set of elements.
- The arrangements (or permutations) of elements may be different.
- If there are repeated elements, the counts of each element must be the same in both arrays.

Examples:

- ***Input:*** *a[] = [1, 2, 5, 4, 0], b[] = [2, 4, 5, 0, 1]*
  Output: *true*
- ***Input:*** *a[] = [1, 2, 5, 4, 0, 2, 1], b[] = [2, 4, 5, 0, 1, 1, 2]*
  Output: *true*
- ***Input:*** *a[] = [1, 7, 1], b[] = [7, 7, 1]*
  Output: *false*

Q4. Given two **singly Linked Lists**, create **union** and **intersection** lists that contain the union and intersection of the elements present in the given lists. Each of the two linked lists contains **distinct** node values.
Note: The order of elements in output lists doesn't matter.

Examples:

- ***Input:***
  head1 : 10 -> 15 -> 4 -> 20
  head2 : 8 -> 4 -> 2 -> 10
  Output:
  Intersection List: 4 -> 10
  Union List: 2 -> 8 -> 20 -> 4 -> 15 -> 10
  Explanation: *In these two lists 4 and 10 nodes are common. The union lists contain all the nodes of both lists.*

- ***Input:***
  head1 : 1 -> 2 -> 3 -> 4
  head2 : 3 -> 4 -> 8 -> 10
  Output:
  Intersection List: 3 -> 4
  Union List: 1 -> 2 -> 3 -> 4 -> 8 -> 10
  Explanation: *In these two lists 4 and 3 nodes are common. The union lists contain all the nodes of both lists.*