**Assignment 1 – Arrays**

**Ans 1. def twoSum (nums, target):**

**num\_dict = {} # Dictionary to store the numbers and their indices**

**for i, num in enumerate(nums):**

**complement = target - num # Calculate the compliment**

**if complement in num\_dict:**

**return [num\_dict[complement], i] # Return the indices if complement is found**

**num\_dict[num] = I # Add the number and its index to the dictionary**

**# If no solution is found, return an empty list or raise an exception**

**return []**

**Ans 2. def removeElement(nums, val):**

**k = 0 # Counter for elements not equal to val**

**for i in range(len(nums)):**

**if nums[i] != val:**

**nums[k] = nums[i] # Move the element to the front**

**k += 1**

**return k**

**Ans3. def searchInsert(nums, target):**

**left, right = 0, len(nums) - 1**

**while left <= right:**

**mid = left + (right - left) // 2**

**if nums[mid] == target:**

**return mid**

**if nums[mid] < target:**

**left = mid + 1**

**else:**

**right = mid - 1**

**return left**

**Ans4. def plusOne(digits):**

**carry = 1**

**n = len(digits)**

**for i in range(n-1, -1, -1):**

**digits[i] += carry**

**carry = digits[i] // 10**

**digits[i] %= 10**

**if carry == 0:**

**break**

**if carry != 0:**

**digits.insert(0, carry)**

**return digits**

**Ans5. def merge(nums1, m, nums2, n):**

**p1 = m - 1 # Pointer for nums1**

**p2 = n - 1 # Pointer for nums2**

**p = m + n - 1 # Pointer for the merged array**

**while p1 >= 0 and p2 >= 0:**

**if nums1[p1] >= nums2[p2]:**

**nums1[p] = nums1[p1]**

**p1 -= 1**

**else:**

**nums1[p] = nums2[p2]**

**p2 -= 1**

**p -= 1**

**# Copy remaining elements from nums2 to nums1 if any**

**nums1[:p2 + 1] = nums2[:p2 + 1]**

**Ans6. def containsDuplicate(nums):**

**seen = set()**

**for num in nums:**

**if num in seen:**

**return True**

**seen.add(num)**

**return False**

**Ans7. def moveZeroes(nums):**

**zero\_ptr = 0 # Pointer for the next position to place a non-zero element**

**# Iterate through the array**

**for i in range(len(nums)):**

**if nums[i] != 0:**

**# Swap the current element with the element at the zero\_ptr position**

**nums[i], nums[zero\_ptr] = nums[zero\_ptr], nums[i]**

**zero\_ptr += 1**

**Ans8. def findErrorNums(nums):**

**num\_set = set()**

**duplicate = -1**

**for num in nums:**

**if num in num\_set:**

**duplicate = num**

**num\_set.add(num)**

**n = len(nums)**

**missing = set(range(1, n + 1)) - num\_set**

**return [duplicate, missing.pop()]**