**DAY-3**

**DATE:13/10/2024**

**1.Sorting lists with various characteristics**

**test\_cases = [[], [1], [7, 7, 7, 7], [-5, -1, -3, -2, -4]]**

**for lst in test\_cases:**

**print(sorted(lst))**

**2. Selection Sort Example**

**arr = [5, 2, 9, 1, 5, 6]**

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

**min\_idx = i**

**for j in range(i + 1, len(arr)):**

**if arr[j] < arr[min\_idx]:**

**min\_idx = j**

**arr[i], arr[min\_idx] = arr[min\_idx], arr[i]**

**print(arr)**

**# Expected Output: [1, 2, 5, 5, 6, 9]**

**3. Optimized Bubble Sort with Early Exit**

**arr = [64, 25, 12, 22, 11]**

**n = len(arr)**

**for i in range(n):**

**swapped = False**

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

**if arr[j] > arr[j + 1]:**

**arr[j], arr[j + 1] = arr[j + 1], arr[j]**

**swapped = True**

**if not swapped:**

**break**

**print(arr)**

**# Expected Output: [11, 12, 22, 25, 64]**

**4. Test Cases for Optimized Bubble Sort**

**test\_cases = [[64, 25, 12, 22, 11], [29, 10, 14, 37, 13], [3, 5, 2, 1, 4], [1, 2, 3, 4, 5], [5, 4, 3, 2, 1]]**

**for arr in test\_cases:**

**n = len(arr)**

**for i in range(n):**

**swapped = False**

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

**if arr[j] > arr[j + 1]:**

**arr[j], arr[j + 1] = arr[j + 1], arr[j]**

**swapped = True**

**if not swapped:**

**break**

**print(arr)**

**# Expected Outputs:**

**# [11, 12, 22, 25, 64]**

**# [10, 13, 14, 29, 37]**

**# [1, 2, 3, 4, 5]**

**# [1, 2, 3, 4, 5]**

**# [1, 2, 3, 4, 5]**

**5. Insertion Sort with Duplicates**

**arr = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3]**

**for i in range(1, len(arr)):**

**key, j = arr[i], i - 1**

**while j >= 0 and arr[j] > key:**

**arr[j + 1] = arr[j]**

**j -= 1**

**arr[j + 1] = key**

**print(arr)**

**# Expected Output: [1, 1, 2, 3, 3, 4, 5, 5, 6, 9]**

**6. Kth Missing Positive Integer**

**arr, k = [2, 3, 4, 7, 11], 5**

**missing, current = [], 1**

**for num in arr:**

**while current < num:**

**missing.append(current)**

**current += 1**

**current += 1**

**while len(missing) < k:**

**missing.append(current)**

**current += 1**

**print(missing[k - 1])**

**# Expected Output: 9**

**7. Finding a Peak Element**

**nums = [1, 2, 3, 1]**

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

**while left < right:**

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

**if nums[mid] > nums[mid + 1]:**

**right = mid**

**else:**

**left = mid + 1**

**print(left)**

**# Expected Output: 2**

**8. Index of First Occurrence of**needle**in**haystack

**haystack, needle = "sadbutsad", "sad"**

**print(haystack.find(needle))**

**# Expected Output: 0**

**9. Substrings in an Array of Strings**

**words = ["mass", "as", "hero", "superhero"]**

**result = []**

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

**for j in range(len(words)):**

**if i != j and words[i] in words[j]:**

**result.append(words[i])**

**break**

**print(result)**

**# Expected Output: ["as", "hero"]**