Python Function Practice

1. Multiply List Elements

Write a function multiply_by_two(numbers) that takes a list of integers and returns a new list where each number is multiplied by 2.

Example: multiply_by_two([1, 2, 3]) returns [2, 4, 6].

Hint: Create an empty list, loop through the input, append num * 2.

2. Count Vowels in a String

Write a function count_vowels(s) that takes a string s and returns the number of vowels (a, e, i, o, u—case insensitive).

Example: count_vowels("hello") returns 2.

Hint: Loop through each character, use lower() and if char in 'aeiou'.

3. Average of Numbers

Write a function average(numbers) that takes a list of integers and returns their average (sum divided by length). Assume list is not empty.

Example: average([1, 2, 3, 4]) returns 2.5.

Hint: Loop to sum, then divide by len(numbers).

4. Replace Spaces with Dashes

Write a function dash_string(s) that takes a string s and returns it with all spaces replaced by dashes ("-").

Example: dash_string("hello world") returns "hello-world".

Hint: Start with empty string, loop through chars; if char == ' ', add '-', else add char.

5. Filter Positive Numbers

Write a function positive_only(numbers) that takes a list of integers and returns a new list with only the positive numbers (> 0).

Example: positive_only([-1, 2, -3, 4, 0]) returns [2, 4].

Hint: Empty new list, loop through input; if num > 0, append it.

6. Find Duplicates in a List

Write a function find_duplicates(lst) that returns a list of numbers that appear more than once in lst (in the order they first appear).

Example: find_duplicates([1, 2, 2, 3, 4, 4, 5]) returns [2, 4].

Hint: Use a dictionary to track seen elements and their counts.

7. Palindrome Checker

Write a function is_palindrome(s) that checks if a string s (ignoring spaces, case, and punctuation) is a palindrome. Return True or False.

Example: is_palindrome("A man, a plan, a canal: Panama") returns True.

Hint: Clean the string first (loop to remove non-alphabets, convert to lowercase), then compare with its reverse.

8. Matrix Diagonal Sum

Write a function diagonal_sum(matrix) that takes a square 2D list (list of lists) and returns the sum of elements on the main diagonal (from top-left to bottom-right).

Example: diagonal_sum([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) returns 15 (1+5+9).

Hint: Loop with range(len(matrix)) and access matrix[i][i].

9. Anagram Groups

Write a function group_anagrams(words) that takes a list of strings and returns a dictionary where keys are sorted strings and values are lists of original words that are anagrams of each other.

Example: group_anagrams(["eat", "tea", "tan", "ate", "nat", "bat"]) returns {"aet": ["eat", "tea", "ate"], "ant": ["tan", "nat"], "abt": ["bat"]}.

Hint: For each word, sort its letters as a key in a dict; append originals to lists.

10. Longest Common Prefix

Write a function longest_common_prefix(strs) that takes a list of strings and returns the longest common prefix among them. If none, return empty string.

Example: longest_common_prefix(["flower", "flow", "flight"]) returns "fl".

Hint: Assume the first string is the base; loop through characters, checking if all strings share it with if conditions.