# Module 6.4 Function Intermediate: Coding Questions with Hints

## Question 1: Create a function that accepts any number of arguments and prints them.

Hint: Use the \*args syntax in the function definition to accept a variable number of arguments.

## Question 2: Write a function that calculates and returns the average of any number of numerical arguments.

Hint: Use \*args to accept multiple arguments and calculate the average inside the function.

## Question 3: Implement a function that takes a list and an arbitrary number of additional arguments, then returns a new list containing only the elements found in both.

Hint: Use \*args for the arbitrary number of arguments and list comprehension or a loop to filter the list.

## Question 4: Define a function that accepts keyword arguments and prints the key-value pairs.

Hint: Use the \*\*kwargs syntax in the function definition to accept keyword arguments.

## Question 5: Create a function that takes two lists of equal length and merges them into a dictionary, where elements from the first list are keys and elements from the second list are values.

Hint: Use zip() to pair elements from both lists and dict() to convert the pairs into a dictionary.

## Question 6: Write a function that uses recursion to calculate the factorial of a number.

Hint: A factorial function calls itself with a decremented number until reaching the base case of 1.

## Question 7: Implement a function that takes a string and returns a dictionary with the count of each character in the string.

Hint: Use a loop to iterate over the string and a dictionary to keep track of counts.

## Question 8: Define a function that filters a list of numbers by removing elements that are not divisible by a given divisor.

Hint: Use list comprehension or filter() to create a list of elements that are divisible by the divisor.

## Question 9: Create a decorator function that prints the name of any function it decorates.

Hint: Define a decorator that takes a function as an argument, wraps it in another function that prints the function’s name, then returns the wrapper function.

## Question 10: Write a function that accepts a variable number of positional and keyword arguments and prints the total number of arguments received.

Hint: Use \*args and \*\*kwargs in the function definition and count the lengths of both.