TP: Python Functions

Say OL

September 27, 2024

Python Functions: Problem Set

This problem set is designed to help you practice writing and using functions in Python. Follow the instructions for each problem and document your progress.

Problem 1: Basic Function Creation

a. Write a function called `greet` that takes a name as a parameter and prints a greeting message.

b. Call the `greet` function with your name as an argument.

Problem 2: Return Values

a. Write a function called `add` that takes two numbers as parameters and returns their sum.

b. Call the `add` function with two numbers and print the result.

Problem 3: Default Parameters

a. Write a function called `power` that takes two parameters: a base and an exponent. Set the default value of the exponent to 2.

b. Call the `power` function with just the base and then with both the base and exponent.

Problem 4: Keyword Arguments

a. Modify the `power` function from Problem 3 to accept keyword arguments.

b. Call the `power` function using keyword arguments for both parameters.

Problem 5: Variable Number of Arguments

a. Write a function called `calculate_average` that takes a variable number of arguments and returns their average.

b. Test the function with different numbers of arguments.

Problem 6: Lambda Functions

a. Create a lambda function that squares a number.

b. Use this lambda function to print the square of a number of your choice.

Problem 7: Nested Functions

a. Write a function called `outer` that contains a nested function called `inner`. The `inner` function should return a greeting message.

b. Call the `outer` function and print the result of the `inner` function.

Problem 8: Function Annotations

a. Write a function called `multiply` that takes two numbers and returns their product. Use function annotations to specify that both parameters should be of type `float`.

b. Call the `multiply` function and display the result.

Problem 9: Decorators

a. Create a simple decorator function called `uppercase_decorator` that converts the output of a function to uppercase.

b. Apply this decorator to a function that returns a string.

Problem 10: Recursion

a. Write a recursive function called `factorial` that takes a number as a parameter and returns its factorial.

b. Call the `factorial` function with a number and print the result.

Optional Challenges

a. **Challenge 1:** Implement a function that generates the Fibonacci sequence up to a given number using recursion.

b. **Challenge 2:** Write a function that counts the number of vowels in a given string.

Submission

Submit your Python files containing the function implementations, along with any necessary test scripts or documentation of your progress.