Arguments in Python User-Defined Functions

Arguments are like inputs that you pass to a function when you call it. They provide the function with the information it needs to perform its task.

Here's a breakdown of different types of arguments in Python functions:

- **Positional arguments:** These are arguments passed to the function in the same order they are defined in the function's parameter list. The function expects a specific number of arguments in a specific order.
- **Keyword arguments:** These arguments are passed using the variable name and an assignment operator (=) when calling the function. This allows you to pass arguments in any order, as long as you use the correct variable names.
- **Default arguments:** These arguments come with a pre-defined value assigned within the function definition. If you don't provide a value for a default argument when calling the function, the default value is used.
- **Arbitrary positional arguments (args):* This allows you to pass a variable number of positional arguments to a function. These arguments are stored as a tuple inside the function.
- **Arbitrary keyword arguments (**kwargs):** This allows you to pass a variable number of keyword arguments to a function. These arguments are stored as a dictionary inside the function.

Return Types in Python User-Defined Functions

The return type refers to the data type of the value a function returns to the calling code. A function can return any data type in Python, including strings, integers, lists, or even custom objects.

- **Explicit return:** You can explicitly use the `return` statement to specify the value the function returns.
- **Implicit return:** If a function doesn't explicitly use a `return` statement, it implicitly returns `None`.

Example: User-Defined Function with Arguments and Return Type

Here's an example of a function that calculates the area of a rectangle and demonstrates arguments and return types:

Python

```
def calculate_area(length, width):
  """This function calculates the area of a rectangle.

  Args:
      length: The length of the rectangle (float).
      width: The width of the rectangle (float).

  Returns:
      The area of the rectangle (float).
  """
  area = length * width
  return area

# Call the function with arguments
rectangle_area = calculate_area(5, 3)
print(f"The area of the rectangle is: {rectangle_area}")
```

In this example:

- `length` and `width` are positional arguments.
- The function calculates the area and returns it (float).

Exercise Questions

1. **Write a Python function that takes two numbers as arguments and returns their sum. Can you make this function handle non-numeric inputs gracefully?**
2. **Write a Python function that checks if a given string is a palindrome (reads the same backward as forward). The function should be case-insensitive.**
3. **Write a Python function that takes a list of numbers as input and returns a new list containing only the even numbers. Use a loop and conditional statement within the function.**

4. **Write a Python function that calculates the factorial of a non-negative integer passed as an argument. Remember, the factorial of a number is the product of all positive integers less than or equal to that number.**
5. **Write a Python function that takes a string as input and returns a new string with all the vowels removed. Can you achieve this without using conditional statements for each vowel?**
6. **Write a Python function that finds the greatest common divisor (GCD) of two positive integers passed as arguments. There are multiple ways to solve this; you can implement the Euclidean algorithm.**
7. **Write a Python function that takes a list of words as input and returns a new list with all the words converted to uppercase. Use list comprehension for a concise solution.**
8. **Write a Python function that checks if a given string is a valid phone number. The function should account for different formatting styles (e.g., with or without hyphens).** (Note: This can be a more complex task depending on the desired level of validation)
9. **Write two Python functions: one that calculates the area of a circle and another that calculates the area of a rectangle. Both functions should take the necessary dimensions as arguments and return the calculated area.**
10. **Write a Python function that uses recursion to print the Fibonacci sequence up to a certain number (passed as an argument). The Fibonacci sequence is where each number is the sum of the two preceding numbers (starting from 0 and 1).**