Function Documentation

Function documentation in Python is typically done using docstrings. A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. It is used to describe the purpose, usage, and parameters of the function. Properly documented functions enhance code readability and provide useful information to developers who may use or maintain the code. Here's an example:

```python
def multiply(a, b):
    """
    Multiply two numbers.

    Parameters:
    - a (int): The first number.
    - b (int): The second number.

    Returns:
    int: The product of a and b.
    """
    return a * b
```

In this example:

- The docstring is enclosed in triple quotes (`"""`).
- The first line is a concise description of the function's purpose.
- The subsequent lines provide more detailed information about the function, such as the parameters it

accepts and the type of values they should have.

- The "Returns" section describes the type of value the function returns.

To access the docstring of a function, you can use the `help()` function or access the `__doc__` attribute:

```python
help(multiply)
# Output:
# Multiply two numbers.
#
# Parameters:
# - a (int): The first number.
# - b (int): The second number.
#
# Returns:
# int: The product of a and b.

print(multiply.__doc__)
# Output:
# Multiply two numbers.
#
# Parameters:
# - a (int): The first number.
# - b (int): The second number.
#
```

Returns:

int: The product of a and b.

```

By convention, it's recommended to follow the [Google Style Python Docstrings](https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html) or [PEP 257](https://www.python.org/dev/peps/pep-0257/) style guide for writing docstrings.

Here's an example of a more complex function with a docstring following the Google Style:

```python
def calculate_discount(original_price, discount_percentage):
    """
    Calculate the discounted price.

    Args:
        original_price (float): The original price before discount.
        discount_percentage (float): The discount percentage to be applied.

    Returns:
        float: The discounted price.
    Raises:
        ValueError: If discount_percentage is less than 0 or greater than 100.
    """
    if discount_percentage < 0 or discount_percentage > 100:
        raise ValueError("Discount percentage must be between 0 and 100.")
```
```

```
    discount_amount = original_price * (discount_percentage / 100)

    discounted_price = original_price - discount_amount


    return discounted_price
```

In this example, the docstring follows a consistent style, providing information about arguments, return values, and potential exceptions that may be raised. Properly documented code like this can be invaluable for developers working with or maintaining the codebase.