■ Project Documentation Index

test_files/sample.py

File Summary

This Python file defines a single function:

- `add(a, b)`*: Returns the sum of two input numbers a and b.*

Example Use Case

```
result = add(2, 3)
print(result)  # Output: 5
```

Functionality

- *The function takes two arguments of any numeric type (e.g., integers, floats).*
- *It returns their sum.*
- *No error handling or input validation is implemented.*

test_files/module1/math_utils.py

File Summary

This Python file defines three basic arithmetic functions:

- `add(a, b)`*: Returns the sum of two numbers a and b.*
- `subtract(a, b)`*: Returns the difference of two numbers a and b.*
- `multiply(a, b)`*: Returns the product of two numbers a and b.*

The file provides a simple implementation of basic mathematical operations, but does not include any error handling or complex logic.

Example Use Cases

- `add(2, 3)` *would return* `5`
- `subtract(5, 2)` *would return* `3`
- `multiply(4, 5)` *would return* `20`

Notes

- *The* `subtract` *function lacks a docstring, which is a good practice to include for clarity and documentation purposes.*
- *The file does not include any main function or execution block, suggesting it is intended to be imported as a module in another script.*

test_files/module2/string_utils.py

File Summary

This Python file contains three simple string processing functions:

1. ***`greet(name)`****: Returns a personalized greeting string for a given `name`.*

2. ***`capitalize_words(text)`****: Takes a string `text` and returns a new string with each word capitalized.*

3. ***`count_words(text)`****: Counts the number of words in a given `text` string and returns the count.*

These functions appear to be utility functions for basic string manipulation and can be used in a variety of applications.

Example Use Cases:

- *`greet("John")` returns `"Hello, John!"`*
- *`capitalize_words("hello world")` returns `"Hello World"`*
- *`count_words("hello world")` returns 2*