Code Understanding Report

Generated: 2025-05-05 23:51:47

This report presents automated insights based on large language models and code analysis tools.

File: pasted_code.py

Summary

- This code defines a decorator `decorator_repeat(func)` that wraps the function `func` and repeats its execution `num_times` times.

The decorator `@wraps(func)` is used to keep the `func`'s original attributes after the decorator is applied. The `wrapper` function is essentially a wrapper around `func` that repeats its execution `num_times` times.

The decorator `decorator_repeat` is then returned as the result of the inner function `repeat`.

The outer function `repeat` - The given code snippet is a simple greeting function written in Python.

The function takes a name as an argument. It then prints out the greeting "Hello, 'name'!"

Docstring

- ### Code: def repeat(num*times): def decorator*repeat(func): @wraps(func) def wrapper(*args,* *kwargs): for _ in range(num*times): result = func(*args, **kwargs) return result return wrapper return decorator*repeat

Docstring:

This code defines a decorator `decorator_repeat` that repeats a function `func` a certain number of times.

Here's how it works:

- The `@wraps(func)` decor
- ### Code: def greet(name): print(f"Hello, {name}!")

Docstring:

def greet(name): """ This function greets the provided name.

```
Args:
    name (str): The name to greet.

Returns:
    str: A greeting message
```

Code Quality

Tool: `pylint`
Issues: 0`

Conclusion

The greet function is a simple function that greets a provided name.