## Overview

This file provides utility functions for various tasks. Its primary function, get\_functions\_and\_classes, uses Python’s Abstract Syntax Trees (AST) to parse Python code and extract metadata about its top-level functions and classes. It also includes other miscellaneous helper functions for demonstration and general use.

## Functions

### get\_functions\_and\_classes()

Parses a string of Python code to extract information about the functions and classes defined within it. This function is robust against syntax errors in the input code, returning an empty dictionary in such cases to prevent analysis failures.

**Parameters:**

* code\_content (str): A string containing the Python code to be analyzed.

**Returns:**

* Dict[str, Dict[str, Any]]: A dictionary where each key is the name of a function or class found in the code. The corresponding value is another dictionary containing details about that object, including its type (‘Function’ or ‘Class’), name, start\_line, and end\_line.

**Example:**

code = """  
class MyClass:  
 def method\_one(self):  
 pass  
  
def my\_function():  
 return True  
"""  
  
objects = get\_functions\_and\_classes(code)  
# objects will be:  
# {  
# 'MyClass': {'type': 'Class', 'name': 'MyClass', 'start\_line': 2, 'end\_line': 4},  
# 'my\_function': {'type': 'Function', 'name': 'my\_function', 'start\_line': 6, 'end\_line': 7}  
# }  
print(objects)

### hello()

A simple utility function that prints the provided argument to the standard output.

**Parameters:**

* name: The value to be printed to the console.

**Returns:**

* This function does not return any value.

### calculate\_fibonacci()

Calculates the n-th Fibonacci number using an iterative approach. This function handles base cases for n <= 0 and n = 1, and then iteratively computes the sequence for larger values of n.

**Parameters:**

* n (int): The position in the Fibonacci sequence (0-indexed).

**Returns:**

* int: The n-th Fibonacci number. Returns 0 for non-positive input.

**Example:**

fib\_10 = calculate\_fibonacci(10)  
# fib\_10 will be 55  
print(f"The 10th Fibonacci number is: {fib\_10}")  
  
fib\_0 = calculate\_fibonacci(0)  
# fib\_0 will be 0  
print(f"The 0th Fibonacci number is: {fib\_0}")