PROGRAMMING ASSESSMENT

1. Write a Python decorator that measures the execution time of a function and logs it. Apply this decorator to a function that performs a computationally expensive task.

import time

import logging

# Set up logging configuration

logging.basicConfig(level=logging.INFO)

def time\_logger(func):

"""Decorator that logs the execution time of a function."""

def wrapper(\*args, \*\*kwargs):

start\_time = time.time()

result = func(\*args, \*\*kwargs)

end\_time = time.time()

execution\_time = end\_time - start\_time

logging.info(f"Function '{func.\_\_name\_\_}' executed in {execution\_time:.4f} seconds")

return result

return wrapper

@time\_logger

def compute\_expensive\_task(n):

"""A computationally expensive task."""

total = 0

for i in range(1, n + 1):

total += i \*\* 2 # Sum of squares

return total

# Example usage

if \_\_name\_\_ == "\_\_main\_\_":

result = compute\_expensive\_task(10\*\*6)

print(f"Result: {result}")

