Python Notes

Decorator Function

* In python, a decorator is a function that modifies the behavior of another function or method without changing its code.
* Decorators are often used for logging, enforcing access control, instrumentation, caching, and more.
* How Decorators Work:
  + A decorator is a function that takes another function as an argument, enhances it or modifies it, and returns the modified function.
  + The @decorator\_name syntax is used to apply a decorator.

Example (Refer Decorator(Example).py)

* Timing\_decorator is a function that takes another function(func) as an argument.
* Inside timing\_decorator, we define a nested function wrapper, which:
  + Starts a timer using time.time()
  + Calls the original function func(\*args, \*\*kwargs), passing any arguments it receives.
  + Stopes the timer.
  + Prints how long the function took to execute.
  + Returns the result of func, ensuring the original function’s return value is not lost.
* Finally, wrapper is returned, replacing the original function with it decorated value.
* @timing\_decorator applies the decorator to predict\_stock\_price, so whenever predict\_stock\_price is called, it is actually calling the wrapper function inside timing\_decorator.
* The function:
  + Prints a message saying it is predicting the stock price.
  + Simulates comutation time by using time.sleep(2), which pauses execution for 2 seconds.
  + Generates a mock predicted price using random.uniform(100, 150), which picks a random float between 100 and 150. The round(…, 2) rounds it to 2 decimal places.
  + Prints the predicted stock price.
  + Returns the predicted price.
* Calling the function for “Tesla”.

Why use a decorator there?

* It automatically times the function execution without modifying the actual function logic.
* Can be reused for timing multiple functions.

Wrapper

* Wrapper is an inner function that replaces the original function while adding extra functionality. It is the function that actually gets executed when you call the decorated function.