Home Assignment <12>: Tester Roles with NumPy and Polymorphism

Learning Objective:

The objective of this assignment is to combine polymorphism with NumPy by creating different tester roles that analyze test execution data differently.

Expected Completion Time:

Best Case: 20 minutes
Average Case: 30 minutes

Assignment Details:

You are simulating different QA roles analyzing test cycle data using NumPy.

Requirements:

a) Create three classes: `ManualTester`, `AutomationTester`, and `PerformanceTester`.
b) Each class should have a method `analyze(data)` that takes a NumPy array of test execution times.

- ManualTester → prints the first 5 test execution times (`data[:5]`).
- AutomationTester → prints the fastest test case (`data.min()`).
- PerformanceTester → prints the 95th percentile execution time (`np.percentile(data, 95)`).

c) Write a function `show_analysis(tester, data)` that calls the `analyze()` method.
d) In the main section:

- Create a NumPy array with at least 12 execution times.
- Create objects of all three tester roles.
- Call `show_analysis()` for each tester object.

Hints to Solve:

- Use NumPy slicing for ManualTester.
- Use `np.min()` for AutomationTester.
- Use `np.percentile()` for PerformanceTester.
- Demonstrate polymorphism by calling the same `analyze()` method on different objects.

Expected Outcome:

Upon completion of this assignment, you should be able to:

- Implement polymorphism with NumPy data analysis.
- Apply the same method (`analyze()`) with different behaviors.
- Use statistical functions in NumPy.
- Relate polymorphism to different QA tester roles.