**Programming Assignment 1 (PA-1): Comparing Algorithms (8% of Final Grade)**

Release Date: January 17 (Friday)  
Due Date: February 4 (Tuesday) @ 11:59 PM, via D2L Dropbox submission

**Student Information:**

* **Full Name:** Thomas Boyko
* **UCID: 30191728**
* **Tutorial Section: T06**

**Part 2 of PA-1: REPORT**

|  |  |
| --- | --- |
| **Item** | **Task** |
| 1. **Timing Table** | Record execution times for all three algorithms at varying N values: 0,1,2,5,10,20,30,40,50. |
| 1. **Performance Chart** | Visualize the timing results in a single line chart to compare algorithm performance. |
| 1. **Complexity Insights** | Relate the observed timing results to the theoretical time complexities of each algorithm |
| 1. **Reflections on AI Tool Use** | Transparent reflection on AI tool use (or not). |

**1. Timing Table**

* Run each algorithm 3–5 times for each N value and compute the average execution time. This ensures results are accurate while keeping the computational workload manageable. If results vary significantly between runs, increase the number of runs for more stable averages.
* Use the table below to record your results:

**Note –** Testing up to N=50 effectively fulfills the pedagogical and practical goals of the assignment, providing meaningful insights into algorithmic efficiency while maintaining feasibility for students and their systems.

**2. Performance Chart**

* Use the provided **PA-1-plotting-results.xls** to generate the chart(s).
* Choose one or both of the following options for visualization:
  + A **single chart** with three distinct lines, one for each algorithm.
  + **Three separate charts**, each focusing on one algorithm.
* Include distinct lines for each algorithm, ensuring proper labels for axes, title, and a legend.
* **X-Axis:** Execution Time (T, in milliseconds).
* **Y-Axis:** Fibonacci Number (N).

**Place your Chart(s) here:**

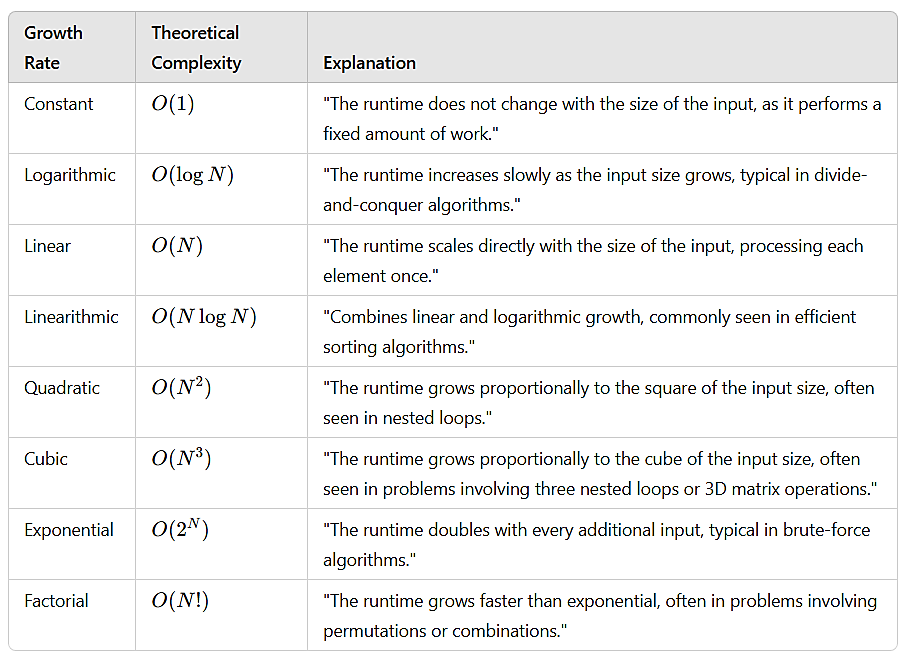
|  |
| --- |
|  |

**3. Complexity Insights**

For each algorithm, explain the observed growth pattern in execution time and justify how it matches the theoretical complexity:

|  |  |  |
| --- | --- | --- |
| Algorithm | Theoretical Complexity | Explanation of observed trends justifying why this growth pattern occurs |
| Algorithm 1:  Recursive | O( 2^n) | Runtime grows incredibly quickly as n increases; each recursive call to fibonacci() calls two more times. |
| Algorithm 2:  Iterative | O(n) | Runtime appears to grow linearly with input size; main loop in program runs linearly with input size |
| Algorithm 3:  Matrix Exponentiation | O(log n) | Runtime appears to grow slightly slower than linearly according to my graph |

Refer to the table below to help determine the appropriate theoretical complexity for each algorithm:



**4. Reflection on AI Tool Use**

**Did you use AI tools for this REPORT?** (Yes/No): No

**If Yes, briefly explain:**

* **How** did the AI tools assist you? (e.g., debugging, brainstorming, generating ideas).

|  |
| --- |
| n/a |

* **Why** did you decide to use them, and **when** during your process?

|  |
| --- |
| n/a |

* **What impact** did using AI tools have on your understanding of the material (e.g., improved clarity, new insights, or reliance concerns)?

|  |
| --- |
| n/a |

**Final Notes**

* Write clearly and concisely. Avoid technical jargon unless necessary.
* Ensure the report is professional, free of grammatical errors, and properly formatted.
* Focus on showcasing your understanding of algorithm performance, not just raw data presentation.