Benchmarking-Sorting-Algorithms

Bubble & Quick Sort

Irving F. Sanchez

Lewis University

Course Number: *SP25-CPSC-34000-002*

Professor: David Nowak

Benchmarking-Sorting-Algorithms

Before diving into performance metrics and algorithm complexity, I want to reflect on the inspiration that helped me approach this project. Furthermore, I’d like to add that I’ve worked on algorithms just like this in previous classes such as my computer engineering class, software engineering, programming languages, and .NET programming—so this isn’t a completely new topic to me. Like my previous work, this wasn’t about getting the bubble sort and quick sort to run—it was about crafting something that someone new to programming could easily walk through step-by-step and feel confident enough to understand. I write my code the way I wish someone would’ve explained things to me when I first started: with clarity, and intentionality. I believe that great code can be both functional and compassionate, dropping fear and intimidation.

The goal of this project was to implement bubble sorting and quick sorting for comparative purposes, then analyze and reflect on their timing performance across datasets increasing size and complexity. To keep the project engaging and relatable, I decided to layer it with a narrative metaphor: an algorithmic tournament judge sorting competitors into brackets. Bubble sort became the slow but methodical judge, manually comparing each contestant. Quick sort, on the other hand, became the clever strategist, dividing the group with pivot-based efficiency.

In designing the program, I also wanted to simulate real-world variability. Rather than hardcoding a single dataset, I integrated three real CSV files—Pokémon statistics, Magic the Gathering token cards, and video game review data. Each offered a different structure and data volume, allowing for authentic comparison across small, medium, and large datasets. The Pokémon data set consisted of every single Pokémon from the first generation up to the current 2025 generation (a total of 1025). The MTG (Magic the Gather) tokens consisted of every single token card made and all their variations from their initial release (a total of seven thousand cards). Lastly, the video game data set consisted of every single ‘rated’ video game from the earliest video game console to the current generation of the platform (13000 in total). Each offered a different structure and data volume, allowing for authentic comparison across small, medium, and large datasets. I added prompts to let the user decide whether to sort by name, rating, power or other attributes—because in the real world, I believe the usefulness of sorting often depends on what you’re trying to understand or optimize.

One surprising insight came from observing just how slowly bubble sort scales. I knew it was , but watching it crawl through a 10,000-item data set was like watching molasses pour uphill. This was a teaching moment in itself—not just in runtime theory but in real, experiential learning. It reinforced why time complexity matters, and why algorithm choice isn’t just academic—it’s practical.

To support user understanding, I designed the interface to be interactive and flexible. Users can run five tests per dataset/iteration, choose sorting conditions (random or semi-sorted), and review the results on a color-coded graph. These aren’t just bells and whistles, they’re scaffolds for engagement and reflection. I also included a “Definitions” section to explain what each sort does in simple easy to read language, because no everyone who runs this program will have taken an algorithms course let alone a programming course.

Like all my previous projects, I layered this one with comments and labeled sections to guide anyone reading the code. Every block is introduced with start/end markers for easy navigation, and every function is built with readability in mind. I still struggle with assignments like these—my self-conscious can feel like a beehive at times, buzzing with what-ifs and “did I forget something?”, or “did I make this clear enough to understand?” But with every project, I try to translate that chaos into clarity. This one was no different. Sorting, at its core, is about making sense of disorder, which is often what happens in my own brain. Perhaps, that’s what all this is about—a discipline of the mind but of the mind of a computer as well and the way it organizes all its data in the most efficient manner.