Jorge Argueta

Southern New Hampshire University

CS-499 Computer Science Capstone 24EW4

Professor Brooke Goggin

31 March 2024

**Introduction**

In the CS-260 course, my ambition was to not only excel academically but also to harness practical skills in computer science. My project, titled "HashTable", originally developed in C++, aimed to deepen my grasp of efficient data management through advanced hash table algorithms. The project transitioned into Python to utilize its robust libraries and was enriched by the adoption of innovative practices and collaborative tools.

**Collaborative Environment**

The "HashTable" project utilized Git as a platform for version control, which served as a vital tool for enabling a collaborative environment. Regular code reviews and mergers facilitated a continuous integration process. I actively sought feedback from diverse peers, and industry practitioners—incorporating their diverse perspectives into the project’s development. This iterative and inclusive approach ensured that decision-making was comprehensive, reflecting a broad spectrum of insights and fostering an environment conducive to collective learning and solution enhancement.

**Communication and Adaptation**

Documentation, commit messages, and discussions were crafted to be clear and concise, ensuring they were accessible to all intended audiences. Detailed written reports were regularly prepared and delivered, showcasing the project's findings and methodological evolutions. Each communication was specifically tailored—technical details were emphasized for academic evaluations, while strategic implications were highlighted for discussions with potential employers, ensuring relevance and engagement.

**Algorithmic Design and Evaluation**

The core of the "HashTable" project involved a meticulous design process evaluating various hashing algorithms and collision resolution strategies, specifically focusing on the use of AVL Trees for collision management. The choice of AVL Trees over simpler linked lists was driven by their O(log n) time complexity for insertions, deletions, and searches, which significantly enhances performance as data scales. Dynamic resizing of the hash table was implemented to maintain operational efficiency, with resizing operations designed to execute at O(n) complexity to minimize downtime and resource consumption.

**Innovation and Implementation**

Python’s built-in data structures and list comprehensions were employed to refine the code’s efficiency and maintainability. Profiling tools were utilized to measure and optimize the performance, ensuring that the hash table's implementation was not only robust but also aligned with industry performance benchmarks. The integration of these tools and techniques demonstrated a commitment to adopting best practices in software engineering and a continuous pursuit of innovation.

**Security Mindset**

Security was a paramount concern throughout the project development. I implemented and evaluated secure hashing algorithms to mitigate risks associated with collision attacks, which are pivotal in maintaining the integrity of data in hash tables. Each algorithm's selection was backed by thorough research and testing against known vulnerabilities, ensuring that the final product was not only efficient but also resilient against potential security threats.

**Conclusion**

The "HashTable" project exemplified a comprehensive educational journey, from theoretical understanding to practical application, embedded in rigorous academic and industry standards. This project was a testament to my ability to learn, adapt, and excel, demonstrating how foundational computer science principles can be effectively applied to create sophisticated, secure, and efficient computing solutions.