Cat Lewin
Professor Brian Hare
CS 441
15 February 2025

# Program 1: LLM & Racket Basics

This first program focused on using LLMs to create a counting sort program utilizing the hash table data structure built in with the Racket language. I used Claude.ai as the generative LLM for this project.

The first implementation of the program used a file-reading frequency counter using hash tables. Though this program was functioning, it failed to actually use the counting sort algorithm. I realized I needed to be more specific in my prompts, but also needed to actually understand the counting sort algorithm. I used ChatGPT to explain the algorithm, read up on the algorithm on geeksforgeeks website, and then further watched CSDojo's youtube video on the subject to understand how the algorithm works. My thinking was: to accurately analyze if the program was working properly, I needed to understand the algorithm in the first place.

With this better understanding of the counting sort, I prompted 2 major modifications: the sorting algorithm & display function. I initially requested Claude to not use the library sort function, causing it to create a merge sort function instead. Realizing my error, I finally specified for Claude to implement a counting sort. The second change was for the display function to better match the output format from the assignment description.

Following a functioning program meeting the requirements, I prompted Claude to create test code. Though the initial implementation had thorough tests, it also had a few errors. After a few prompts, Claude was able to create a functioning test suite that ran smoothly with the main program.

Generally, Claude generated fair quality code that integrated well together. The initial problems with the program were a result of my failure to be specific in my prompts and my misunderstanding of the counting sort. When prompted with specific goals (e.g. use the counting sort, fix this system error), Claude was able to modify the previously generated code accurately while maintaining a functioning program. Claude was quick to adapt to new requirements, diagnosed problems and handled errors. I fixed the program through prompting Claude directly and did not need to manually intervene to modify code.

The main shortcoming of the generated program is for large lists (such as Data-7.txt) with a wide range of values. Though able to process the file, the program takes a few minutes to do so. That said, this shortcoming seems more reflective of the count sort algorithm which relies on the minimum and maximum elements. This causes the time complexity to worsen with larger ranges.

My largest takeaways from this project were being impressed with Claude's capabilities in code generation, and clarity on how to more effectively use AI in the generation process. The expression "garbage in, garbage out" seems apt.