Unix Project Report

The biggest hurdle I faced was extracting the usage data in a form that can be converted to the graphical bars. Bash requires that all data be an integer for math to be done, so all of the measurements had to be parsed as such. This was further complicated by the need to use different tools–*top*, */proc/meminfo/*, and *df*–to access the different resource usages, so each access method needed to be bespoke for each resource. A more minor hurdle was creating an efficient loop that could create the graphical bars from the usage data.

The *awk* and *grep* commands proved invaluable for the data access. Each data point could be retrieved using *grep* to grab the line with the data and *awk* to choose the part of the line that had the actual value. For the CPU data, I had to use the *printf* function inside of *awk* to explicitly cast the CPU usage (100-*idle*), which is in the form *XX.X*, to an integer. For memory usage, total and available memory had to be grabbed with *grep* and *awk* as separate variables to calculate used memory. For disk usage, *grep* and *awk* were used again to obtain *Use%* of total system disk space, but *sed* had to be used to remove the *%* symbol to store the value as an integer. To make an efficient loop, rather than create a *count* variable to use in a *while* loop, I found that *for i in (seq 1 $filled)* can be used. This creates a list from 1 to our usage variable, which then can be looped through.

If I were to expand on this project, I could add several quality-of-life features to the display. This might include colors for faster understanding of the information, such as a red tint for high usage. I could also monitor total system usage of individual programs, or on the flipside, usage of individual CPU cores or disk partitions. Finally, I might also log data to a *CSV* file to see and process usage over time.