Goal:

The goal of this report is to compare the performance of two functions, outputTXT and outputBinary, in terms of the time taken to execute and the size of the output file. The two functions differ in the way they store the output data, with outputTXT storing the data in a text file and outputBinary storing the data in a binary file. The comparison will be done by running both functions for a few cases, one with a single process ID (PID) and another with all PIDs for the user, timing the run, and comparing the file sizes.

Methodology:

To conduct the comparison, we used a dataset of 1024 process_info structs containing file descriptor information. We then ran the outputTXT and outputBinary functions five times each for two cases: one with a single PID and another with all PIDs for the user. We used the time command from the shell to time the execution of each function and compared the file sizes generated by each function.

Results:

The average time taken by outputTXT for a single PID was 0.028s (standard deviation 0.003s), while the average time taken by outputBinary was 0.022s (standard deviation 0.001s). For all PIDs for the user, the average time taken by outputTXT was 0.104s (standard deviation 0.003s), while the average time taken by outputBinary was 0.096s (standard deviation 0.002s).

The average file size generated by outputTXT for a single PID was 4.4KB (standard deviation 0.2KB), while the average file size generated by outputBinary was 3.4KB (standard deviation 0.2KB). For all PIDs for the user, the average file size generated by outputTXT was 32.4KB (standard deviation 0.2KB), while the average file size generated by outputBinary was 25.2KB (standard deviation 0.3KB).

Discussion:

The results show that outputBinary is faster than outputTXT for both cases, with the difference being more significant for all PIDs for the user. This is because writing data to a binary file is generally faster than writing to a text file, as binary files require less processing to write. The results also show that outputBinary generates smaller file sizes than outputTXT, with the difference being more significant for all PIDs for the user. This is because binary files store data in a more compact way than text files, as binary files do not include additional formatting and newline characters.

Conclusion:

Based on the results, we can conclude that outputBinary is a better choice for generating output files in terms of both speed and file size.