

Problem 1

Findings

- Both multiprocessing and multithreaded merge sorts are way slower than single threaded version
- This can be attributed to time taken to context switch and forking new process or creating a thread is much longer than the parallelism gains if any
- For $n > 2000$ multi process mergeSort fails to sort properly as forking of new process fails. To remedy this I have put a fallback to normal mergeSort in such cases. For much larger case $n = 1e6$, there is segmentation fault possibly due to new array allocations in merge function run out of memory.
- Multithread mergeSort also tends to fail for cases of $n > 10000$ on my system, as system cannot allocate more threads. I have set a fallback to normal mergeSort in such cases

