Report

Multiprocessing vs Multithreading for Implementing multiprogramming:

Multiprocessing: Multiple process is something that can run two or more programs simultaneously with more central processing units

Multithreading: A thread is a flow of execution through the process code, with its own program counter, system registers, and a stack. Which executes independently but concurrently shares the resources.

Now, Let us consider both the cases

Consider Implementing multiprogramming with Process:

- Multiprocessing increase the consistency of the system.
- It can actually completes process in sequential manner.
- It is more efficient when there are multiple system user and needs multiple processing units
- For the given question, we should implement multiprogramming for few sentences to generate its histogram
- When we consider multiple processes, we actually not need multiple processes to complete this task because there is no need of one or more central processing units, Though it is the same task that we have to do, we just need to do it on a single file and single dataset.
- There are some disadvantage like process are really heavy when compared to threads and its complex to implement for this kind of tasks

Consider Implementing multiprogramming with Threads:

- Multithreading can simplify code, increase efficiency, thread creation is light
- Kernels are generally multithreaded. Threads provide a way to improve application performance through parallelism. While one thread is blocked and waiting, a second thread in the same task can run.
- Also, Multithreading uses fewer resources. All threads can share same set of open files, child processes
- When we consider multiple threads, This is the best way to implement this task when compared to multiprocessing
- There are many advantages of implementing multithreading when compared with implementing multiprocess for this task
- We can create multiple threads for this task and can be executed simultaneously, Also resources will be shared among all those threads

Conclusion:

Therefore, I will be Implementing MultiThreading for this task