Operating System

Homework #3 97521036

Question 3 Explanation:

The main question is about the difference between **multi-processing** and **multi-threading** programming, as we know the common concept of these two ways of programming is to do multiple jobs and works efficiently, so we can save much more time than just running all of the jobs on one process or thread.

But let's discuss about the difference:

- In multi-threading we can even have only a single process and have multiple code segments which we want them to be executed concurrently, but in multi-processing we have more than single process.
- Multi-processing can improve the computational power but multi-threading creates computing threads on one process.
- The creation of process (fork()) may need much more resources than creating a thread.
- We can have threads running in parallel on one process but multi-processing can execute multiple processes simultaneously.
- A good point of multi-threading is this fact that a thread is lighter than a process.
- A good point about multi-processing is that is more understandable than multi-threading and it gives us the ability to kill the process if it's not needed but a thread is not killed and we could just exit from it(pthread_exit(NULL)) [as we had it in the code!].
- Thread needs protection in critical regions for common resources, so it's more complicated to handle than process.

In my opinion these were the noticeable differences that I could find them out, and the obvious thing that makes multi-processing look better for a programmer is the simplicity of using it than thread because it doesn't need protection, which in thread it's a hassle to handle it, it's just created and given the instructions and the function it should execute and then it's killed! Really easy and enjoyable to work with but on the contrary is thread!