Indian Institute of Technology Palakkad
Department of Computer Science and Engineering
Operating Systems - Jul to Nov 2020
03 November 2020

Instructions:

- *You should do the assignment in the same group as you did lab4 and lab5*
- *Create a folder named "lab7" in the same repository you created previously*
- *Do the assignment, commit and push your changes when you are done.*
- *Submission deadline: 10-Nov, 2020 23:59 hrs.*

In this assignment, you have to simulate the race from the very popular fable where a Hare and a Turtle enters a contest to see who is faster. You have to simulate the race using two different implementations (of the simulator).

1. Process-based simulator:
    a. Create 4 processes: **`Hare`**, **`Turtle`**, **`God`**, and **`Reporter`**
    b. Use UNIX/Linux pipes for Inter-Process Communication (IPC).
    c. The **`Reporter`** process starts the race, and also it will print the positions of the hare and the turtle on the screen when the race is in progress.
    d. When the race starts, the **`Hare`** and the **`Turtle`** process starts to increment a counter (the counter indicates the current position of the hare and turtle, and therefore, the **`Hare`** process should increment its count relatively faster than the **`Turtle`** process).
    e. If the hare is sufficiently ahead of the turtle, then the **`Hare`** process should sleep for a random amount of time (to simulate what happens in the fable).
    f. **`God`** process can reposition the hare and the turtle any time before the race finishes. NOTE: intervention from **`God`** is optional. Therefore, it is possible that the **`God`** process may choose to not intervene at all. Even in such a scenario, the race must continue.
    g. **`Reporter`** process should print current positions of hare and turtle on the screen (using either a GUI, or ncurses, or simple terminal output).

2. Thread-based simulator:
    a. Implement the same (as described above) using threads instead of processes.
    b. Use pthreads library for creating and managing threads, and mutexes for synchronization between threads.