Simulated Operating Systems
Phase 1

CPU

Implement the instruction executions in CPU. For some instructions, you only need to simulate some register level operations, like the micro-instructions we have discussed in our lectures. But there are also some more complex instructions (refer to the table of instructions, including print and sleep), which will put the process to a wait state. For print, an IO request to theterminal should be issued for the process. For sleep, a timer request should be issued for the process so that the process can get awaken after the sleep time is over. During instruction execution, it is necessary to access memory. Retrieving the instruction and the data has been done in fetch_instruction, except for the store instruction because it needs to write to memory, not to read from memory. You need to take care of the memorywrite when you code for the store instruction. You also need to check the return value (could be normal, error, or page fault) and perform proper actions for various return values.

When there is a page fault or error or when the program executes a special instruction, such as print and sleep, the process can no longer continue execution and the execution status of the process needs to be properlyupdated to let the system know what subsequent processing steps should be taken.

An interrupt may impact the execution of the running process. One of them is time quantum expiration (tqInterrupt). You need to write the interrupt handling code to handle tqInterrupt. For tqInterrupt, the current running process should be stopped and placed back to the ready queue. Actions such as queue insertion, processstatus update, etc., should be coded accordingly.

Process Manager

In the process manager (process.c), you are supposed to implement the main part of execute_process(). Thisinvolves performing context switch, invoking cpu_execution, and checking the process execution status in PCBand deciding what to do next. You also need to keep track of the execution time (the number of instructions executed) and add it to the total time used by the process (PCB[?]->timeUsed).

You have learnt context switch and the code for context switch functions is also left for you to implement. Essentially, you need to select the correct set of registers to save and the correct set for restoring.