Assignment: Process

Use this form to complete your assignment. Students may work alone or with a partner. To begin, go to the OSTEP: Three Easy Pieces website and follow the instructions to download/access the homework code.

Then do all of the homework problems listed at the end of Chapter 4

Enter your responses into the form fields below.

Your email address (**hermann@pdx.edu**) will be recorded when you submit this form. Not you? Switch account

* Required

Your Name *

Hermann Yepdjio

Partner's Name (if you worked with a partner)

N/A

Response for problem #1 *

The CPU utilisation should be 100% because both processes (5:100 and 5:100) specify that chances that each of their instructions is a CPU instruction are 100%. Therefore, one of the processes will be in "running" state while the other one will be in "ready" state waiting for the first one to both complete and move to "done" state. Then the second process will move to "running" state and then "done" state when it is done running its instructions. This means the CPU will be busy the whole time since there's no I/O instructions

Response for problem #2 *

It takes 10 clock ticks to complete both processes (4 for the 1st process + 6 for the second)

Response for problem #3 *

if we switch the order of processes, both processes will take 6 clock ticks. The order matters because if the process that contains I/O instructions is run first, while it is in "blocking" state, the following process can move to "running" state and use the CPU since it is available.

Response for problem #4 *

When we run the two processes (-l 1:0,4:100 -c -S SWITCH_ON_END), the second one waits for the first one to move to "done" before moving to "Running" state. This make both process take 9 time ticks to complete

Response for problem #5 *

when using SWITCH_ON_IO, the second process move to "running" state and start using the CPU as soon as the first process is in "Blocking" state waiting for I/O. This makes both processes to complete in 6 time ticks

Response for problem #6 *

When running this combination of processes (./process-run.py -l 3:0,5:100,5:100,5:100 -S SWITCH_ON_IO -I IO_RUN_LATER -c -p) the second process starts using the CPU as soon as the first moves to "Blocking" state waiting for I/O. When the first process is done waiting, it moves to "Ready" state and waits for all the other processes to finish using the CPU before running again. This causes all the processes to complete in 27 time ticks

Response for problem #7 *

When using IO_RUN_IMMEDIATE, the first process moves to "Running" state whenever it is not waiting for "I/O". When it needs to wait for I/O, it moves to "Blocking" state and the other processes run until they are done or until the first process is done waiting and is ready to use the CPU again.
running a process that just completed an I/O again is a good idea because it reduces the overall time taken by all the processes to complete. In fact, this process will probably run for a small amount of time, then will have to complete an I/O again and while it is completing it, the other processes can run again. Doing this saves a lot of time.

Response for problem #8 *

When running -s 1 -l 3:50,3:50,
- the first process has 1 CPU instruction and 2 I/O instructions.
- the second process has 3 CPU instructions
- therefore the first process starts running its CPU instruction then an I/O then moves to "Blocking" state allowing the second process to move from "Ready" to "Running" state and run all its 3 instructions. Then the fist process finishes running its instructions while waiting for I/O when necessary.
-Both processes take a total of 12 time ticks to complete
- Using -I IO_RUN_LATER vs -I IO_RUN_IMMEDIATE does not make any difference because process 2 moves to "Done" state while process 1 is still waiting for I/O
- When using -S SWITCH_ON_IO, the trace is the same as process 2 starts using the CPU whenever process 1 is waiting for I/O (So, total of 12 time ticks)
- When using SWITCH_ON_END, process 2 waits for process 1 to move to "Done" state before it starts using the CPU, which causes both process to take a longer time to complete (14 time ticks)

Submit