CSSE 332 -- Operating Systems

CPU Scheduling I

Name: Solution Key

Question 1. (5 points) You are the lead designer of a new CPU process scheduler in the latest version of the RHIT Linux operating system. Unlike a regular general-purpose operating system, you know beforehand that the system you are designing only runs three processes: P_1, P_2, and P_3, and that we only have a single CPU.

To make things simple, we will make the following assumptions about these processes:

1. P_1, P_2, and P_3 run for 5 `ms` each.
2. All three processes are ready to run at boot time.
3. You cannot interrupt a process that is actively running on the CPU.
4. Once a process runs on the CPU, it will never block during its runtime. In other words, all three processes are *CPU-intensive* and never sit idle.

Based on the above assumptions, describe a policy by which you determine which process to run at what time. Make sure to note the time (in `ms`) when each process starts and when it ends.

Solution: A possible solution here given our assumptions is to just run each process until it completes, then pick another one and run that one, and so on.

Question 2. (5 points) Given you scheduling policy, for each process, calculate the time difference between when a process completes its execution, and the time when it arrives to your system (recall that they all arrive at time 0 for our current system). We refer to this value as the *turnaround* time for each process. Finally, calculate the average turnaround time for all three processes.

Average turnaround time: ______________________________

Process	**Arrival time**	**Completion time** (`ms`)	**Turnaround time** (`ms`)
P_1	0		
P_2	0		
P_3	0		

Question 3. (5 points) Assume now that we drop our first assumption, i.e., that all three of the processes run for 5 `ms` each. Each process can now run for an arbitrary time. In other words, our assumptions now are:

1. ~~P_1, P_2, and P_3 run for 5 `ms` each.~~
2. All three processes are ready to run at boot time.
3. You cannot interrupt a process that is actively running on the CPU.
4. Once a process runs on the CPU, it will never block during its runtime. In other words, all three processes are *CPU-intensive* and never sit idle.

Describe a scenario where your proposed solution in **Question 1** would fail to run all three of the processes, or would cause significant change to the average turnaround time.

Question 4. (5 points) Under the same assumptions, what would be a way to solve the above problem? Describe your proposed scheduling policy and then calculate the new average turnaround time.

Process	**Arrival time**	**Completion time** (`ms`)	**Turnaround time** (`ms`)
P_1	0		
P_2	0		
P_3	0		

Average turnaround time: ______________________________

Question 5. (5 points) We will now relax our second assumption, which is that all three processes are ready to start at boot time (i.e., at time `0`). Processes can now arrive at different times during the lifetime of our system. For your reference, here are our updated assumptions:

1. ~~P_1, P_2, and P_3 run for 5 `ms` each.~~
2. ~~All three processes are ready to run at boot time.~~
3. You cannot interrupt a process that is actively running on the CPU.
4. Once a process runs on the CPU, it will never block during its runtime. In other words, all three processes are *CPU-intensive* and never sit idle.

Describe a scenario in which the problem you uncovered in **Question 3** resurfaces. Again, calculate the turnaround time for each process and then the average turnaround time for all three processes. It might help you to think of the worst-case possible for arrival times of the processes.

Process	**Arrival time**	**Completion time (`ms`)**	**Turnaround time (`ms`)**
P_1			
P_2			
P_3			

Average turnaround time: ______________________________

Question 6. (5 points) Without relaxing our third assumption, there is really not much we can about the previous problem. Therefore, we will go ahead and relax that assumption and give the scheduler the ability to perform *context switching*. We will allow the scheduler to pause a process and remove it from the CPU to allow another process to start running. Processes removed from the CPU are stored in a data structure, and it is the job of the scheduler to implement the scheduling policy and decide which process gets to run when.

For your reference, here are our updated assumptions:

1. ~~P_1, P_2, and P_3 run for 5 ms each.~~
2. ~~All three processes are ready to run at boot time.~~
3. ~~You cannot interrupt a process that is actively running on the CPU~~.
4. Once a process runs on the CPU, it will never block during its runtime. In other words, all three processes are *CPU-intensive* and never sit idle.

If our goal is to **minimize** turnaround time, i.e., finish our processes as fast as possible. Describe a scheduling policy that would solve the problem revealed in **Question 5** and minimize the turnaround time. As usual, calculate the turnaround time for each process and the average turnaround time for your processes.

Process	**Arrival time**	**Completion time (ms)**	**Turnaround time (ms)**
P_1			
P_2			
P_3			

Average turnaround time: ______________________________

Question 7. (5 points) In addition to the turnaround time, we will introduce a new metric, the *response time*. The response time is defined as the difference between when a process first runs (i.e., the first time it goes on the CPU) and its arrival time. The response time is not concerned with when a process terminates.

For your policy described in the previous question, calculate the response time for each process as well as the average response time over all three.

Process	**Arrival time**	**Time of first run** (`ms`)	**Response time** (`ms`)
P_1			
P_2			
P_3			

Average response time: ______________________________

Question 8. (5 points) If our goal is to minimize response time (rather than turnaround time), how would your scheduling policy change? Suggest a new scheduling policy that can achieve that goal, and calculate the average response time and average turnaround time under the new policy.

Process	**Arrival time**	$T_{\texttt{firstrun}}$ (`ms`)	$T_{\texttt{completion}}$ (`ms`)	$T_{\texttt{turnaround}}$ (`ms`)	$T_{\texttt{response}}$ (`ms`)
P_1					
P_2					
P_3					

Average turnaround time: ______________________________

Average response time: ______________________________