Investigating Process Scheduling

Learning Objectives

At the end of this lab you should be able to:

1. *Describe first-come-first-served (FCFS) process scheduling method*
2. *Describe priority-based process scheduling method*
3. *Explain the difference between non-pre-emptive and pre-emptive priority scheduling mechanisms*
4. *Demonstrate the use of round-robin process scheduling method in sharing the CPU between different processes*

Tutorial Exercises

Initial Preparation

CPU-OS Simulator simulates the functions of a typical modern CPU and a typical Operating System (OS) both of which are supported by the inbuilt assembler and compiler. To start the simulator, double-click on the simulator icon. This starts up the CPU simulator's main window. In order to be able to do the following exercises, you need to enter a program which the OS simulator can run. To do this you need to go to the compiler by clicking on the **COMPILER…** button in the **Advanced** tab. Enter the following program in the edit window and compile.

```
program SimpleLoop
    for n = 1 to 40
        p = p + 1
    next
end
```

When the above program is successfully compiled you'll need to load it in the CPU's memory. You do this by clicking on the **LOAD IN MEMORY…** button. This will then take you back to the CPU simulator. To view the OS simulator's main window first select the **Advanced** tab and then click on the **OS 0…** button.

LO 1: *Describe first-come-first-served (FCFS) process scheduling method.*

Now that the program code is in simulator's memory it can be run. To do this you need to manually create one or more instances of it (real OS does this automatically). First make sure the **FCFS** scheduling method is selected in the **Policies** tab. Next select the **Process** tab and using the **CREATE NEW PROCESS** button create the processes in Table 1, in the order they are listed, making sure correct priorities are selected:

Process Id	Priority
P1	2
P2	4
P3	3

Table 1

You should see the processes listed in the **READY PROCESSES** view. This view represents the Ready Queue. The process at the top of the list is the head of the queue and the last process in the list is the tail of the queue.

Now, OS simulator is ready to run the processes. To do this first drag the **CPU Speed** slider to a position around 60 (the number is displayed as you drag the slider). This is required so that the processes run slowly in order to allow easy observations. Click on the **START** button. You should now see a process in the **RUNNING PROCESSES** view. This view shows the currently running process.

Next, create a fourth process, P4, with priority 1 and once this process is created fill in the required information in Table 2 below:

Running Process	**Waiting Processes** (first entry is the head of the queue)
Pid 3	Pid 1

Table 2

Move the **CPU Speed** slider to the **Fast** position (i.e. top of the slider) and wait until ALL processes run to normal completion (this may take few seconds).

Make a brief comment in the space below regarding the way FCFS scheduling works:

FCFS works by executing the processes in the order that it is given. The first process is given priority while the last one is executed last

LO 2: ***Describe priority-based process scheduling method.***

For this exercise you need to select a different scheduling method. To do this select the **Policies** tab. Then select the **Priority (static)** option and the **Non-preemptive** option. Next, using the **CREATE NEW PROCESS** button create the four processes listed in the Table 1, one after the other, making sure correct priorities are selected.

Now repeat the same procedure as in the first exercise and enter the results in the Table 3 below:

Running Process	**Waiting Processes** (first entry is the head of the queue)
p4	p1
	p3
	p2

Table 3

Move the **CPU Speed** slider to the **Fast** position (i.e. top of the slider) and wait until ALL processes run to normal completion (this may take few seconds).

Make a brief comment in the space below regarding the way non pre-emptive priority scheduling works:

Non pre-emptive priority scheduling works by if there is a process with lower priority running and you add a new process with higher priority enters, the CPU will put that at the head of the queue until the current process stops the execution.

LO 3: ***Explain the difference between non-pre-emptive and pre-emptive priority scheduling mechanisms.***

In this exercise you'll again use priority scheduling method as in the above exercise but this time you need to select the **Pre-emptive** option in the **Policies** tab. Once again, using the **CREATE NEW PROCESS** button create the four processes listed in the Table 1, one after the other, making sure correct priorities are selected.

Now repeat the same procedure as in the first exercise and enter the results in the Table 4 below:

Running Process	Waiting Processes (first entry is the head of the queue)
p1	p2
	p3
	p4

Table 4

Move the **CPU Speed** slider to the **Fast** position (i.e. top of the slider) and wait until all processes run to normal completion (this may take few seconds).

Make a brief comment in the space below regarding the way pre-emptive priority scheduling works summarising how the two priority scheduling methods differ:

If a process is executing at the moment. If a new process enters the queue, the CPU will give the control over to the new process and halt or suspend the current executing process.

LO 4: ***Demonstrate the use of round-robin process scheduling method in sharing the CPU between different processes.***

For this exercise you need to change the scheduling method to **Round Robin (RR)** in the **Policies** tab. Also select the priority option **None** meaning process priorities will not be used. Next you need to select **RR Time Slice** of 5 ticks from the drop-down list. This is used by the OS simulator to decide how long a process will be allowed to run before it is returned to the ready queue. Finally, tick the **Suspend on state change** check box under the **RUNNING PROCESSES** view. This suspends the process as soon as it is put into the running state to facilitate observations.

Now create the processes as specified in Table 5 below (To specify process lifetime select the **Secs** option and enter the time in seconds in the **Lifetime** box. You need to do this before creating the process by clicking on the **CREATE NEW PROCESS** button)

Process Id	Process lifetime in ticks
P1	10
P2	7
P3	20

Table 5

Move the **CPU Speed** slider to the **Fast** position (i.e. top of the slider) and click the **START** button. As the processes move from Ready to Running state they are suspended. Make a note of the running process and click on the **RESUME** button to re-start the suspended running process. Repeat this each time making note of the next running process until all processes run to normal completion. Enter the sequence of the running processes in Table 6 below:

Elapsed tick count	5	5	5	5	2	10	15	20	
Process Id	p1	p2	p3	p1	p2	p3	p3	p3	

Table 6

Make a brief comment in the space below regarding the way round robin scheduling works:

Round robin scheduling works by going through the processes and going through the amount of ticks in each count.

Note: The OS simulator maintains a running log which can be viewed to obtain information on the history of process states. To view this log select the **View** tab in the OS simulator window and click on the **VIEW LOG…** button. This is for your information only.