Operating Systems

U.C. Irvine Division of Continuing Education
EECS X490.11

Assignment #2

Question #1: Explain the difference between preemptive and nonpreemptive scheduling?(5 points)

Question #2: Suppose the following processes arrive for execution at the times indicated. Each process will run for the amount of time listed. In answering the questions, use nonpreemptive scheduling, and base all decisions on the information you have at the time the decision must be made. (10 points)

Process	Arrival Time	Burst Time
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1

1. What is the average turnaround time for these processes with the FCFS scheduling algorithm?
2. What is the average turnaround time for these processes with the SJF scheduling algorithm?

Question #3: Consider the following set of processes, with the length of the CPU burst time given in milliseconds: (20 points)

Process	Burst Time	Priority
P_1	2	2
P_2	1	1
P_3	8	4
P_4	4	2
P_5	5	3

The processes are assumed to have arrived in the order P_1, P_2, P_3, P_4, P_5, all at time 0.

a. Draw four Gantt charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF, non- preemptive priority (a larger priority number implies a higher priority), and RR (quantum = 2).

b. What is the turnaround time of each process for each of the scheduling algorithms in part a?

c. What is the waiting time of each process for each of these scheduling algorithms?

d. Which of the algorithms results in the minimum average waiting time (over all processes)?

Question #4: Assume that a system has multiple processing cores. For each of the following scenarios, describe which is a better locking mechanism (10 points)

1. Spinlock
   or
2. Mutex lock(where waiting processes sleep while waiting for the lock to become available):

scenarios:

- The lock is to be held for a short duration.
- The lock is to be held for a long duration.

Question #5: Use the semaphore(sem.acquire() and sem.release()) in the proper location in the following python code to solve shared data problem. (10 points)

```
import threading
import time

n = 0
array = [0]*10
sem = threading.Semaphore()

def proc(num):
  global n
  while True:
    n = n+1
    if n > 9:
        break
    array[n] = n
    print ("Thread {}: {}".format(num,array))
    time.sleep(0.25)

t1 = threading.Thread(target = proc, args=[1])
t2 = threading.Thread(target = proc, args=[2])
t1.start()
t2.start()
t1.join()
t2.join()
```

Expected output should be like this:

Thread 1: [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
Thread 2: [0, 1, 2, 0, 0, 0, 0, 0, 0, 0]
Thread 1: [0, 1, 2, 3, 0, 0, 0, 0, 0, 0]
Thread 2: [0, 1, 2, 3, 4, 0, 0, 0, 0, 0]
Thread 1: [0, 1, 2, 3, 4, 5, 0, 0, 0, 0]
Thread 2: [0, 1, 2, 3, 4, 5, 6, 0, 0, 0]
Thread 2: [0, 1, 2, 3, 4, 5, 6, 7, 0, 0]
Thread 1: [0, 1, 2, 3, 4, 5, 6, 7, 8, 0]
Thread 2: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]