Chapter 5 (12.5 pts)

Practice Exercises

5.4(a)

5.5(a)

**5.4** Consider the following set of processes, with the length of the CPU burst

time given in milliseconds:

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, nonpreemptive

priority (a larger priority number implies a higher

priority), and RR (quantum = 2).

**5.5** The following processes are being scheduled using a preemptive, roundrobin

scheduling algorithm.

Process Priority Burst Arrival

*P*1 40 20 0

*P*2 30 25 25

*P*3 30 25 30

*P*4 35 15 60

*P*5 5 10 100

*P*6 10 10 105

Each process is assigned a numerical priority,with a higher number indicating

a higher relative priority. In addition to the processes listed below,

the system also has an **idle task** (which consumes no CPU resources and

is identified as *Pidle*). This task has priority 0 and is scheduled whenever

the system has no other available processes to run. The length of a

**Practice Exercises 253**

time quantum is 10 units. If a process is preempted by a higher-priority

process, the preempted process is placed at the end of the queue.

1. Show the scheduling order of the processes using a Gantt chart.

Chapter 6 (12.5 pts)

Practice Exercises

6.2(provide more than that from the publisher’s answer, use structure of semaphore and its list from PPT and book to explain)

**6.2** What is the meaning of the term ***busy waiting***? What other kinds of

waiting are there in an operating system? Can busy waiting be avoided

altogether? Explain your answer.

Chapter 7 (12.5 pts)

Explain in the Readers-Writers Problem, how writer process and reader process use the Semaphores of both rw\_mutex and mutex to achieve process synchronization.

Chapter 8 (12.5 pts)

Practice Exercises

8.3 (use Allocation matrix, Max matrix, Need matrix and available matrix to explain)

**8.3** Consider the following snapshot of a system:

***Allocation Max Available***

*AB C D AB C D AB C D*

*T*0 0 0 1 2 0 0 1 2 1 5 2 0

*T*1 1 0 0 0 1 7 5 0

*T*2 1 3 5 4 2 3 5 6

*T*3 0 6 3 2 0 6 5 2

*T*4 0 0 1 4 0 6 5 6

Answer the following questions using the banker’s algorithm:

a. What is the content of the matrix ***Need***?

b. Is the system in a safe state?

c. If a request from thread *T*1 arrives for (0,4,2,0), can the request be

granted immediately?