### Problem 1

25

### Problem 2

15

### Problem 3

15

### Problem 4

20

### Problem 5

25

<table>
<thead>
<tr>
<th>Problem 1</th>
<th>Problem 2</th>
<th>Problem 3</th>
<th>Problem 4</th>
<th>Problem 5</th>
<th>Total Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>100</td>
</tr>
</tbody>
</table>

STUDENT NAME:

STUDENT ID:
Problem 1: (25 points)
(a) (4 points) Even when a process is swapped out to disk, some information must be maintained for it in the process table, which is always kept in the primary memory by the kernel. Identify at least four items of information that must always be maintained in the primary memory.

(b) (3 points) In many of the modern shared-memory multiprocessor systems, processes contend for a critical section using a spinlock, which is implemented using an instruction such as test-and-set. Should a process perform file IO operation inside a critical section implemented using a spinlock? Briefly justify your answer.
(c) (3 points) Identify at least three important differences between a condition variable in the monitor model and a counting semaphore.

(d) (3 points) Consider a system with three jobs with service times 10, 20, and 30. Suppose that they are executed on a system with processor-sharing based scheduling discipline. What will be the average turnaround time for these three jobs?
(e) (3 points) Consider a multiprogrammed computer system with four processes. Suppose that each process spends 50% of its time in performing IO. What is the CPU utilization?

(f) (3 points) Identify at least two conditions under which operating systems may boost the priority of a process temporarily.
(g) (3 points) Under what conditions the use of multithreaded server (with kernel level threads) is justifiable? Under what conditions would you use a single-threaded server?

(h) (3 points) Identify at least three conditions which can cause the kernel to switch CPU from the currently running process to another ready-to-run process.
Problem 2 (15 points): Consider a system with two processors. There are five jobs – A, B, C, D, and E – waiting in the ready queue to be processed. Their respective service times, as they appear in the queue (starting with the job at the head of the queue) are 5, 4, 3, 2, and 1 seconds. Assume that the jobs arrived at the same time, and a job can be executed on any of the available processors.

(a) (6 points) For the following two scheduling policies, draw the timeline diagrams for each of the two processors, showing the jobs being executed on these two processors.
- FCFS
- Shortest Job First

(b) (6 points) Determine the average turnaround time for each of the two scheduling policies in part (a) above.

(c) (3 points) Compute the average waiting time for the two cases in part (a).
Continue answer for Problem 2...
Problem 3 (15 points): Consider a real-time system consisting of two periodic tasks. Assume that both these tasks require only the CPU and do not perform any I/O. Task $A$ has period of 50 seconds and requires 25 seconds of CPU time. Task $B$ has period of 75 seconds and requires 30 seconds of processing time. For each task, its period is also the deadline. Assume that both these tasks are put in the ready-queue at time 0.

1. Is the condition for the rate monotonic (RM) scheduling satisfied? Is a schedule possible using RM-based static priorities? If yes, show a schedule, otherwise show a case where a task deadline is missed with RM based scheduling. (Note: For any calculations, $\sqrt{2} = 1.414$)

2. Now, suppose that we upgrade the CPU of this system with a CPU which is 25% more powerful than the original one, i.e. the new CPU is 1.25 times faster than the original one. Is the RM scheduling condition satisfied? Is it possible to schedule these tasks using the rate monotonic (RM) scheme? If yes, give a schedule.
Continue answer of Problem 3...
Problem 4: (20 points)

Consider three communicating processes consisting of a main process $M$, a reader process $R$, and a printer process $P$. These processes execute concurrently and asynchronously such that $R$ reads the data and passes it to process $M$, which performs some operation on each data item and passes it to the printer process $P$ which prints each processed data item. **Only one buffer of size $N$ is given**, which is implemented using an array of size $N$ (buffer indices are 0 through $N-1$).

Using semaphores write code for these three processes to properly synchronize their access to the shared buffer.

Process $R$ repeatedly puts a data item in the buffer if there is space available.

Process $M$ repeatedly gets the next available data item to be processed. It gets the item and after processing it puts it in at the same place in the buffer from where it obtained the item.

Process $P$ repeatedly removes the next processed item from the buffer for printing. It removes the item from the buffer and prints it.
continue answer for problem 4...
Problem 5 (25 points):

In this problem you are asked to write a Hoare monitor called RestroomMonitor to properly synchronize the use of a shared restroom facility. When a woman is using the facility, other women can enter and use the facility, but no men can enter it, and vice versa.

This monitor will have the following four methods: woman_wants_to_enter, woman_leaves, man_wants_to_enter, man_leaves. The monitor will keep track of the status of the restroom such as empty, women-using, men-using, and appropriately block a user from entering the facility.

The protocol for using the facility by a man would look as follows:

RestroomMonitor.man_wants_to_enter();
    Enter and use the restroom facility;
    Leave the facility;
RestroomMonitor.man_leaves();

Write the code for this RestroomMonitor showing the following:

• All state variables and condition variables required in your solution
• Implementation of the monitor procedure woman_wants_to_enter.
• Implementation of the monitor procedure man_leaves.
continue answer to problem 5...