Roll Number: _____ Name_____ Group_____

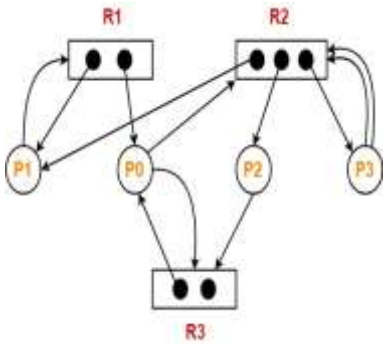
THAPAR INSTITUTE OF ENGINEERING &TECHNOLOGY, PATIALA
Department of Computer Science & Engineering
Operating System (UCS303) – Mid Semester Examination
Date: 03/10/2024 & Time: 11:00AM MM: 30 & MT: 120 Min

Faculty Name: Dr. Garima Singh, Dr. Shashank Sheshar Singh, Dr. Rahul Nijhawan, Dr. Javed Imran Dr. Vaibhav Pandey, Dr. Sumit Varshney, Ms. Shivani Goswami, Dr. Payal

Attempt/Answer all sub-parts like (a), (b), (c) for each question in one place. Do mention Page No. of your attempt on the front page of the answer sheet. *Assume missing data (if any). Show all intermediate computations properly.*

<table>
<tr><th colspan="2">Ques. No</th><th>Questions</th><th>Marks</th><th>CO</th><th>BL</th></tr>
<tr><td rowspan="2">Q1.</td><td>(a).</td><td>Draw a labelled diagram to showcase the various states of a process. Also, label the suitable transition arrow for Short-term, Medium-tem, and Long-term schedulers for their specific work.</td><td>[2+1]</td><td>CO2</td><td>L2</td></tr>
<tr><td>(b).</td><td>What is a deadlock? Explain the deadlock characteristics.</td><td>[2]</td><td>CO2</td><td>L2</td></tr>
<tr><td rowspan="2">Q2.</td><td>(a).</td><td>With suitable diagrams, explain inter-process communication models. Also, explain direct and indirect message-passing inter-process communication models to establish logical links with related issues.</td><td>[2+1]</td><td>CO2</td><td>L2</td></tr>
<tr><td>(b).</td><td>List and explain two system calls for each process management and file management.</td><td>[1+1]</td><td>CO1</td><td>L2</td></tr>
<tr><td>Q3.</td><td></td><td>Consider the below table of five processes under Multilevel Feedback Queue scheduling. Queue1 uses a round-robin scheduling algorithm having a time quantum of 2 ms, Queue2 also uses a round-robin scheduling algorithm with a time quantum of 4ms, and Queue3 uses the pre-emptive shortest job first (SRTF) algorithm for process scheduling.

The priority of Queue1 is the highest, and the priority of Queue 3 is the smallest (Queue1>Queue2>Queue3), and there is preemptive priority scheduling between the queues to avoid larger waiting times. A new process enters Queue1 first and moves to a lower level queue if it does not finish in a given time quantum; if at Queue2 a process doesn't get complete in a given time quantum it moves to queue3. If, at any point, the process waiting time is greater than or equal to 11 (WT >= 11 ms), it must be upgraded to a higher priority queue (1 level up). The waiting time of a process is considered after its arrival in the current ready queue (i.e., waiting in previous queues are ignored). Find out the average waiting time.
<table>
<tr><th>Process</th><th>Arrival Time</th><th>Burst Time</th></tr>
<tr><td>P1</td><td>2</td><td>10</td></tr>
<tr><td>P2</td><td>3</td><td>8</td></tr>
<tr><td>P3</td><td>7</td><td>12</td></tr>
<tr><td>P4</td><td>12</td><td>9</td></tr>
<tr><td>P5</td><td>8</td><td>8</td></tr>
</table></td><td>[5]</td><td>CO2</td><td>L5</td></tr>
<tr><td rowspan="2">Q4.</td><td>(a).</td><td>Differentiate between single-threaded processes and multi-threaded processes with suitable diagrams.</td><td>[2]</td><td>CO2</td><td>L4</td></tr>
<tr><td>(b).</td><td>Consider a uniprocessor system with ‘n’ processes in the ready queue. Round-robin scheduling with time quantum ‘x’ is used for process scheduling. Assume each process requires ‘kx’ seconds to complete, and the context switch takes ‘s’ seconds. At what</td><td>[3]</td><td>CO2</td><td>L5</td></tr>
</table>

<table>
<tr><td></td><td></td><td>time will the first process complete the execution? (Assume all the variables are integers).</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Q5.</td><td></td><td>Draw an annotated parent-child tree structure [including the return value of each fork ()] for the code snippet given. [Here, 1, 2, 3… denotes line number]
a. Comment on the total count of occurrences for “OS Mid Semester”, and “2024” using annotated parent-child tree, when x=1.
b. Find out the total count of occurrences for “OS Mid Semester”, and “2024”, when x=2. [You can use an annotated parent-child tree when x=1 for explanation]</td><td>1. int main()
2. {
3. fork();
4. for(int i=0; i<x; i++)
5. {
6. if ((fork() && ! fork()))
7. {
8. printf(“OS Mid Semester”);
9. fork();
10. }
11. }
12. fork();
13. printf("2024");
14. return 0;
15. }</td><td>[3+2]</td><td>CO1</td><td>L3</td></tr>
<tr><td>Q6.</td><td>(a).</td><td>Consider the given code snippet to represent various sections of process boundary (text, data, heap, and stack) in memory. Complete the given table based on the program's storage in the various sections of the process memory.
<table><tr><th>Process Section</th><th>Program Components</th></tr><tr><td>Text</td><td></td></tr><tr><td>Data</td><td></td></tr><tr><td>Heap</td><td></td></tr><tr><td>Stack</td><td></td></tr></table></td><td>int x=5;
int add (int i, int j)
{
return i+j;
}
int main()
{
static int y=2;
int z=1;
int* ptr = (int*)malloc(n * sizeof(int));
int sum = add(z,y);
printf("sum:", %d);
return 0;
}</td><td>[2]</td><td>CO2</td><td>L2</td></tr>
<tr><td></td><td>(b).</td><td>Consider the given resource allocation graph (RAG) with four processes (P0, P1, P2, and P3) and three resources (R1, R2, and R3). Resources are multi-instance types, as given in RAG. Find the safe sequence using the safety algorithm if the given RAG is deadlock-free; otherwise, find the additional minimum resources required for each type to avoid deadlock.</td><td> 
</td><td>[3]</td><td>CO2</td><td>L4</td></tr>
</table>

Marks Distribution