USN: | | | | | | | | | | | |

# RV University

## School of Computer Science

### B.Tech (H) Degree Examination – Dec 2023

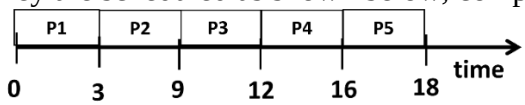
**Semester : 3**

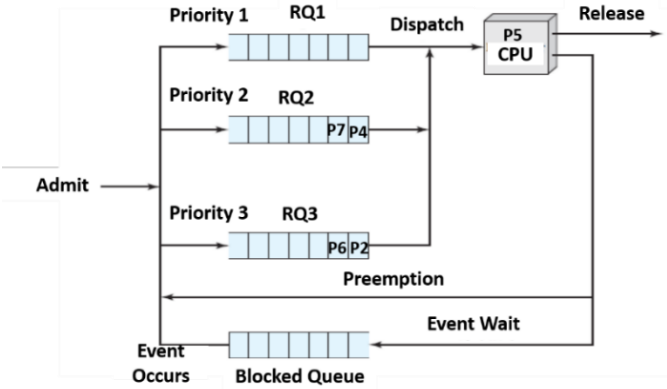
**Course Code : CS2100**

**Course Title : Operating Systems and Systems Software**

**Duration : 2 Hours** **Max. Marks: 30**

**Instructions to students (If any): 1.** Answer all the questions in Part A. **2.** Part-B any 10 out of 12 questions can be answered. The best 10 answers will be considered for evaluation. **3.** No further clarifications will be provided during the examination. You can mention your assumptions, if any.

| Q. No | Short descriptive answers: Part A<br>(10 Questions * 1 mark) – 10 Marks | Marks |
|---|---|---|
| 1 | Is the below statement true or false? Justify your answer briefly.<br>Some of the privileged instructions can be executed only in the kernel mode. | **1** |
| 2 | Give one possible reason for a process which is in the Blocked state to move to Ready state. | **1** |
| 3 | Assume that all the below five processes have become ready at time zero. If they are scheduled as shown below, compute the average waiting time.<br>  | **1** |
| 4 | **double myArr[8][4];** // assume row major<br>If the above double array is stored in the memory starting from the base address 48 (decimal number), what will be the address of the element myArr[3][2]? | **1** |
| 5 | What happens when the below instructions are executed?<br>**system("ls -l");**<br>**printf("I have executed system() call\n");** | **1** |
| 6 | Assuming all the processes have completed their execution at time unit 19. Which process was preempted many times before its completion? Give the number of preemptions of that process as well.<br>  | **1** |
| 7 | What happens when a process executes **sleep(10)** system call? | **1** |
| 8 | What are the two basic operations or functions that can be performed on a Semaphore? Briefly explain about both. | **1** |
| 9 | Briefly explain what happens when two processes are livelocked. | **1** |
| 10 | Briefly explain about the spinlock. | **1** |

<table>
<tr><th>Q. No.</th><th>Problems and long descriptive answers: Part B<br>Answer any 10 out of 12 (10 Questions * 2 marks) – 20 Marks</th><th>Marks</th></tr>
<tr><td>11</td><td>Draw the Gantt chart if the SRTF scheduler is running in the system.<br><table><tr><th>Process Id</th><th>Arrival time</th><th>Burst time</th></tr><tr><td>P1</td><td>0</td><td>9</td></tr><tr><td>P2</td><td>1</td><td>4</td></tr><tr><td>P3</td><td>2</td><td>9</td></tr></table></td><td>2</td></tr>
<tr><td>12</td><td>What is a Process Control Block? Explain about any two entries in them.</td><td>2</td></tr>
<tr><td>13</td><td>Briefly explain about the shared memory and how it is used by processes.</td><td>2</td></tr>
<tr><td>14</td><td>Explain any two responsibilities of memory management system.</td><td>2</td></tr>
<tr><td>15</td><td>Answer the below questions if the following are the properties of a memory management system: Assume a single level of paging.<br>1. Page size = 1 KB<br>2. Logical address is 16 bits wide<br>3. Physical address is 14 bits wide<br>a. What is the total size of Physical memory in the system?<br>b. What is the max no. of entries would a page table of a process have?<br>c. How many frames would there be in the physical memory?<br>d. What is the width of the offset field in the logical address?</td><td>2</td></tr>
<tr><td>16</td><td>Ready queues of three priority levels with processes in them are shown below. Within the same priority level, LIFO is the scheduling used. Assume that the currently running process P5 creates two new processes P8 followed by P9 both with Priority 2.<br>Assume, a non-preemptive scheduler is running in the system.<br>Assume Priority 1 is the highest, with Priority 1 > Priority 2 > Priority 3<br>Which is the next process that would run on the CPU after P5 terminates?<br> 
</td><td>2</td></tr>
<tr><td>17</td><td>Briefly explain about the binary and counting semaphores.</td><td>2</td></tr>
<tr><td>18</td><td>Briefly explain how using dynamic libraries help in reducing the size of the executables.</td><td>2</td></tr>
<tr><td>19</td><td>Explain briefly about the below function and its parameters.<br>pthread_create(&thread_id, NULL, thread_function, (void *)message);</td><td>2</td></tr>
</table>

| | | |
|---|---|---|
| 20 | How many times below messages would be printed? Order does not matter.<br>int main(){<br>pid_t id;<br>printf("Message 0\n");<br>id = fork();<br>if(id == 0){<br>printf("Message 1\n");<br>return 0;<br>}<br>fork();<br>fork();<br>printf("Message 2\n");<br>return 0;<br>} // end of main() | **2** |
| 21 | Explain along with msgs output by this program. Assume that it runs fine.<br>void *threadFn1(void *arg);<br>void *threadFn2(void *arg);<br>int main(){<br>void *result1; void *result2;<br>pthread_t id1, id2;<br>pthread_create(&id1, NULL, threadFn1, "Hi once");<br>pthread_create(&id2, NULL, threadFn2, "Hi again");<br>pthread_join(id1, &result1);<br>printf("Result Msg: %s\n", (char*)result1);<br>pthread_exit(0);<br>}<br>void * threadFn1(void *arg){<br>printf("Thread 1 created with a msg: %s\n", (char*)arg);<br>sleep(3);<br>pthread_exit("Bye ...");<br>}<br>void * threadFn2(void *arg){<br>printf("Thread 2 created with a msg: %s\n", (char*)arg);<br>sleep(5);<br>printf("Slept for 5 seconds.\n");<br>exit(0);<br>} | **2** |
| 22 | Give the output of this program.<br>int myArr[2][4] = {1, 2, 3, 4, 5, 6, 7, 8};<br>int main(){<br>printf("Q22: val1 = %d val2 = %d\n", myArr[0][3], myArr[1][3]);<br>return 0;<br>} | **2** |

**Signature of Paper Setter** **Signature of Scrutiniser**

**Signature of the Dean**

## Answer Keys:

**Q1. Kernel mode is provided in the CPU to enable OS to execute some of the privileged instructions (related to interrupts, memory mgt, etc.).**

**Q2. A process could be in Blocked state for various reasons (waiting for an event to happen or a file to be opened or timer to be triggered, etc.) when the condition a process is waiting for happens, OS moves the process from the Blocked state to Ready state.**

**Q3. Avg waiting time = (0 + 3 + 9 + 12 + 16) / 5 = 40/5 = 8time units**

**Q4. double myArr[8][4]; m = no. of rows = 8, n = no. of columns = 4**

**myArr[i][j] = BA + (i*n + j) * SIZE // SIZE = 8 for char BA = 48, n = 4**
**The address of myArr[3][2] in decimal is: 48 + (3 * 4 + 2) * 8 = 48 + (12 + 2)*8 = 160**

**Note: No partial marks.**

**Q5. ls -l is executed, giving the details of files in the current directory. – 0.5 marks**

**The printf() statement is executed after the above command. – 0.5 marks**

**Q6. P3 got preempted the most. The number of preemptions before its completion was 3. 0.5 marks for P3 and the correct no. of preemptions as 3 – 0.5 marks**

**Q7. sleep(10) system call makes the process wait for 10 seconds before it continue with running and executing the next instructions after the sleep() system call.**

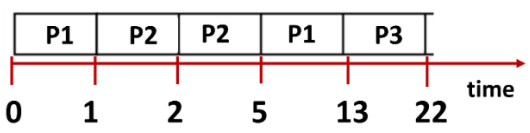
**Q8. sem_wait() and sem_signal() or sem_post() are the functions that can be performed on a semaphore – 0.5 marks each (explanation is optional)**
**sem_wait()** – The calling process gets blocked if the semaphore is not available. If available, the process acquires the semaphore and continues with the execution.
**sem_signal() or sem_post()** – When a process completes the use of a Semaphore it was holding, it releases it back by calling this function on the semaphore.
**sem_init() –** Those who have gone thorough the code and understood it will write this too.
**Note**: Even if **sem_init()** is not mentioned, do not reduce any marks. The explanation should be closer to what is given here, need not be the same.

**Q9. The states of the processes involved in the livelock constantly change but none of them will be able to progress or useful work is done.**

**Q10. A spinlock is a lock which causes a thread trying to acquire it to simply wait in a loop or busy wait ("spin") while repeatedly checking if the lock is becoming available. It comes out of it as soon as the lock is available.**
**It is used wherever critical sections are expected to be short and moving the processes to blocked state is a waste of time instead they are allowed to busy wait.**

## Part B

**Q11. If the SRTF scheduler is running the Gantt chart is as below:**




**Q12. PCB is maintained for every active process in the system – 1 mark**

**PCB may contain one of the details: PID, priority, context information, any files being handled, etc. – 1 mark**

**Q13. Shared memory is the mechanism used for sharing the contents on a physical memory to between two different processes. Their individual logical addresses could be different for different processes which have access to the shared memory, but they are mapped to the same physical location in the main memory. – 1.5 marks**
**They need to be accessed with controlled access mechanism like mutex() to avoid race conditions, when they are being accessed by multiple processes. – This will be mentioned by those who have gone through the code and understood it. – Full marks if this mentioned.**

**Q14. Ref Session 6B that explains about the responsibilities or solutions provided by the Memory management system of the OS.**

**Q15. 1. Page size = 1 KB**

**2. Logical address is 16 bits wide**

**3. Physical address is 14 bits wide**

**a. What is the total size of Physical memory in the system? 16 KB.**

**b. What is the max number of entries would a page table of a process have? $2^6$ or 64 entries.**

**c. How many frames would there be in the physical memory? 16 frames.**

**d. What is the width of the offset field in the logical address? 10 bits.**

**Q16. ANS P9:**
**Short explanation is required justifying your answer.**
The next Process that would run after P5 terminates is: **P9** (this would be the process at the head of the highest priority ready queue (Priority 2) Since P5 is creating P8 with Priority 2 first, it will be in the head of Priority 2 Queue and later P9 will be added ahead of P8 in the same Priority 2 queue because of LIFO) So, P9 will be picked up. **Note**: Since it is a non-preemptive scheduler, no time-slice need to be assumed.

**Q17. Binary Semaphores can be either 0 or 1, 0 value: Busy and 1 value: Free**
**Whereas for counting semaphores, the initial value (indicating free condition) need not be one. It is initialized with a value which is equal to the number of resources that can be used in parallel by different processes.**

**Q18. When dynamic libraries are used, the library is not built with the executable thus reducing the size of the executable. Because of this reason loading time of the executable is less. But it takes more time when the module within the dynamic library is called by the running process for the first time, because it needs to be loaded by the OS dynamically.**

**Q19. pthread_create(&thread_id, NULL, thread_function, (void *)message);**
**It creates a thread by giving the entry function as thread_function that accepts a void * as the input parameter. It writes the thread id of the newly created thread, into the pointer**

**passed as the first parameter. The second parameter is the attribute of the thread created, if it is passed as NULL, a thread is created with the default parameters.**

**Q20. Order can be arbitrary; the number of messages needs to be correct. Relatively reduce marks based on the correctness.**

**Message 0 – once**
**Message 1 – once**
**Message 2 – four times**

**Q21. Give 0.5 marks each if the messages are given correctly. Order does not matter.**

**Thread 1 created with a msg: Hi once**
**Thread 2 created with a msg: Hi again**
**Result Msg: Bye ... → This message may be printed and may not get printed too.**
**Slept for 5 seconds.**
**Note: If the above detail in blue is mentioned full marks, otherwise 1.5 marks.**

**Q22. Output of this program, one mark each for the correct answer.**

**Q22: val1 = 4 val2 = 8**

*******