

**KIIT Deemed to be University**

**Online End Semester Examination(Spring Semester-2021)**

**Subject Name & Code:**OS **Applicable to Courses: B.Tech**

**Full Marks=50** **Time:2 Hours**

**SECTION-A(Answer All Questions. Each question carries 2 Marks)**

**Time:30 Minutes (7×2=14 Marks)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Q. No** | **Question Type** | **Question** | **Answer Key** |
| Q.No:1 | MCQ | Arrange the following Memory/Storage Device in increasing order of proximity to the CPU.   1. Registers, Electronic disk, Main Memory(RAM),   Cache.   1. Registers, Cache,   Electronic disk, Main Memory(RAM).   1. Registers, Cache,   Main Memory(RAM),  Electronic disk.   1. Cache,   Registers  Main Memory(RAM),  Electronic disk. | (c) |
|  | MCQ | Arrange the following Memory/Storage device in increasing order of speed of execution.   1. Magnetic tapes, Electronic disk, Optical disk,   Magnetic disk.   1. Magnetic tapes, Electronic disk, Magnetic disk Optical disk, 2. Magnetic tapes, Optical disk, Magnetic disk, Electronic disk. 3. Magnetic tapes, Optical disk, Electronic disk Magnetic disk. | (c) |
|  | MCQ | A memory storage which is both volatile and non-volatile.  a)Electronic disk  b) Optical disk  c) RAM  d)Cache | Any option will get full mark  (a, b, c, d) |
|  | MCQ | Arrange the following Memory/Storage device in increasing order of cost.   1. Magnetic tapes, Registers, Optical disk, Magnetic disk. 2. Magnetic tapes, Optical disk, Magnetic disk, Registers 3. Magnetic tapes, Cache, Magnetic disk, Registers. 4. Magnetic tapes, Cache, RAM, Registers. | (b) |
| Q.No:2 | MCQ | Which of the following is a valid process state transitions?   1. Ready – Ready 2. Ready – Waiting 3. Waiting – Ready 4. Waiting – Running | (c) |
|  | MCQ | Which of the following is an invalid process state transitions?   1. Ready – New 2. Ready – Waiting 3. Waiting – Running 4. All of the above | (d) |
|  | MCQ | CPU scheduler involves the following queue   1. Device queue 2. Ready queue 3. Both of above 4. None of above | (b) |
|  | MCQ | Mid-term scheduler involves the following transition  a)Ready-Running  b)Running-Waiting  c)Both of above  d)None of above | (d) |
| Q.No:3 | MCQ | Which of the following transition relates to non-preemption  a) Running-Ready  b) Running-Waiting  c)Ready-Running  d) Waiting-Ready | (b, c, d) |
|  |  | If the fork() is called ‘n’ times by the parent process, then the number of total processes generated are:   1. 2n 2. n^2 3. n^2/2 4. 2^n | (d) |
|  |  | Which of the following scheduling algorithms may produce starvation:  a)FCFS  b)Priority  c) Only a  d) Both a and b | (b) |
|  |  | Increasing time quantum in Round Robin (RR) scheduling implies:  a)RR behaves like FCFS  b)Higher average turn around time  c)RR behaves like SJF pre-emptive  d)Only b  e)None of the above | (a) |
| Q.No:4 |  | Given a Resource allocation graph (RAG) with multiple instance multiple resources, choose the correct statement:   1. A cycle in RAG guarantees deadlock. 2. A cycle in RAG means no deadlock. 3. A cycle in RAG may or may not guarantee deadlock. 4. Absence of cycle may guarantee no deadlock. | (c) |
|  |  | Given a Resource allocation graph (RAG) with single instance multiple resources, choose the correct statement:   1. A cycle in RAG guarantees deadlock. 2. A cycle in RAG means no deadlock. 3. A cycle in RAG may or may not guarantee deadlock. 4. Absence of cycle may guarantee no deadlock. | (a) |
|  |  | Given a Resource allocation graph (RAG) ‘m’ resource types and ‘n’ processes, choose the correct statement:   1. RAG algorithm is more efficient than Banker’s algorithm by a factor of ‘m’. 2. RAG algorithm is more efficient than Banker’s algorithm by a factor of ‘1/m’ types. 3. RAG algorithm is more efficient than Banker’s algorithm by a factor of ‘m/n^2’. 4. RAG algorithm is more efficient than Banker’s algorithm by a factor of ‘n^2/m’. | (a) |
|  |  | Given a Wait for graph (WFG) ‘m’ resource types and ‘n’ processes, choose the correct statement:   1. WFG algorithm is more efficient than Deadlock detection algorithm by a factor of ‘1/m’. 2. WFG algorithm is more efficient than Deadlock detection algorithm by a factor of ‘m’. 3. WFG algorithm is more efficient than Deadlock detection algorithm by a factor of ‘n^2/m’. 4. WFG algorithm is more efficient than Deadlock detection algorithm by a factor of ‘m/n^2’. | (b) |
| Q.No:5 |  | Given two atomic operations on semaphore p(),v() choose the correct option:   1. p(): s++; v(): s++ 2. p(): s--; v(): s++ 3. p(): s++; v():s-- 4. p(): s--; v(): s--   The problem of busy waiting in semaphore is solved by following functions:   1. wait(); sleep() 2. wait(); signal() 3. block(); wakeup() 4. block(); wait() | (b)  (c) |
|  |  | Given that P and Q are two processes on semaphores S, Q such that:  Wait(S) Wait(Q);  Wait(Q) Wait(S);  Choose the correct option:   1. The above sequence will lead to a deadlock. 2. The above sequence may lead to a deadlock. 3. No deadlock will happen. 4. None of the above. | (b) |
|  |  | A solution for priority inversion may be:   1. Priority acquiring 2. Priority reduction 3. Both of the above 4. None of the above | (d) |
|  |  | Let processes P and Q access a shared variable S with critical section. Pick the correct statement:   1. P, Q modify S simultaneously 2. Q then P may modify S. 3. P then Q may modify S 4. Both (b) and (c) 5. None of the above | (d) |
| Q.No:6 |  | Given a logical memory of size 16KB and page size of 4B. If the physical memory has a total of 8 bits, (assume byte addressable memory), choose the most appropriate option:   1. Data insufficient 2. 64 frames, 256B RAM, 12 page table entries 3. 64 frames, 256B RAM 4. 64 frames, 256B RAM, 256 KB page table size 5. Address mapping not possible | (C) |
|  | MCQ | Given a logical memory of size 16KB and page size of 4B. If the physical memory has a total of x bits, (assume byte addressable memory), choose the most appropriate option:   1. Data insufficient 2. 2^(x-2) frames, 256B RAM, 12 page table entries 3. 2^(x-2) frames, 2^x B RAM, 2^x KB page table size 4. 64 frames, 2^x B RAM, 2^x KB page table size 5. Address mapping not possible | No answer available. So, Grace mark to all options  (a, b, c, d) |
|  | MCQ | Given a logical memory of size x KB and page size of 4B. If the size of physical memory and that of page table is 256B (assume byte addressable memory), choose the most appropriate option:   1. No. Of pages is 4K 2. Size of logical memory is 16KB 3. Data insufficient 4. Both (a), (b) | (a, b, c, d) |
|  |  | Given that the total number of TLB accesses is 100 and the total number of page table accesses is 20. If the TLB access time is 20 ns, page table and RAM access time is 100 ns each then there is:   1. 45% slowdown in memory-access time 2. 40% slowdown in memory-access time 3. 50% slowdown in memory-access time 4. Data insufficient | (b) |
| Q.No:7 | MCQ | External fragmentation may be dealt with schemes:   1. Only Compaction 2. Paging 3. Only Paging, Segmentation 4. Compaction, paging, segmentation | (a, b) |
|  |  | Perform FIFO with the following page sequence: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 having a frame size of 4. The number of page faults:  a)9  b)10  c)11  d) None of the above | (b) |
|  |  | Perform FIFO with the following page sequence: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1 having a frame size of 3. The number of page faults:  a)15  b)11  c)13  d)10 | (a) |
|  |  | Perform LRU with the following page sequence: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1 having a frame size of 3. The number of page faults:  a)15  b)12  c)13  d)10 | (b) |

**SECTION-B(Answer Any Three Questions. Each Question carries 12 Marks)**

**Time: 1 Hour and 30 Minutes** **(3×12=36 Marks)**

|  |  |
| --- | --- |
| **Q No** | **Question** |
| **Q.No:8** | 1. Explain the difference between batch, multiprogramming, time-sharing and distributed operating system. [2] 2. Consider process arrival as given below where N = right most significant digit of your Roll No.( ex:- for Roll No. 18085**4**, N=**4**):  |  |  |  |  | | --- | --- | --- | --- | | Process | CPU Burst Time(ms) | Arrival Time | Priority | | A | 4 | 0 | 2 | | B | 3 | 3 | 3 | | C | 6 | N | 6 | | D | 5 | 5 | N | | E | 1 | 15 | 4 |   Calculate the following for *priority (non preemptive)* and *round robin* (time quantum = 2 ms) CPU scheduling algorithm: [10]   1. Average waiting time 2. Turnaround time for each process 3. Order of completion   (*hints:-higher digits indicate higher priority*)  Solution:   1. The users of a batch operating system do not interact with the computer directly. Each user prepares his job on an off-line device like punch cards and submits it to the computer operator. Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time. Time-sharing or multitasking is a logical extension of multiprogramming. Processor's time which is shared among multiple users simultaneously is termed as time-sharing. Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly. The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as loosely coupled systems or distributed systems. 2. Non-preemptive priority (higher numbers indicate more priority, N = 4)   Process Start (ms) End (ms) Wait (ms) Response (ms) Turnaround (ms)  A 0 4 0 0 4  C 4 10 0 0 6  D 10 15 5 5 10  E 15 16 0 0 1  B 16 19 13 13 16  Average Waiting Time: 18ms/5 = 3.6 ms  Order of Completion: A, C, D, E, B  Round Robin (Quantum = 2 ms)  Time Process Queue  0-1 A -  1-2 A -  2-3 A -  3-4 A B  4-5 B C  5-6 B C, D  6-7 C D, B  7-8 C D, B  8-9 D B, C  9-10 D B, C  10-11 B C, D  11-12 C D  12-13 C D  13-14 D C  14-15 D C  15-16 C D, E.  (Giving D priority over E as D has arrived earlier, they both have same remaining time and priority, however, at this point)  16-17 C D, E  17-18 D E  18-19 E -  Waiting Times  Process Waiting Time (ms) Turnaround Time (ms)  A 0 4  B 1 + 4 = 5 8  C 2 + 3 + 2 = 7 13  D 3 + 3 + 2 = 8 13  E 3 4  Average waiting time = 23 ms/5 = 4.6 ms  Order of Completion: A, B, C, D, E |
| 1. What is the drawback of preemptive priority scheduling algorithm? Explain method to solve it. [2] 2. Consider process arrival as given below where N = right most significant digit of your Roll No.( ex:- for Roll No. 18085**4**, N=**4**):  |  |  |  |  | | --- | --- | --- | --- | | Process | CPU Burst Time(ms) | Arrival Time | Priority | | A | 4 | 0 | 2 | | B | 3 | 1 | 3 | | C | 6 | 3 | 6 | | D | 5 | N | N | | E | 1 | 4 | 4 |   Calculate the following for *priority (non preemptive)* and *round robin* (time quantum = 2 ms) CPU scheduling algorithm: [10]   1. Average waiting time 2. Turnaround time for each process 3. Order of completion   (*hints:-higher digits indicate higher priority*)  Solution:  (a). Here, are cons/drawbacks of priority scheduling   * If the system eventually crashes, all low priority processes get lost. * If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time. * This scheduling algorithm may leave some low priority processes waiting indefinitely. * A process will be blocked when it is ready to run but has to wait for the CPU because some other process is running currently. * If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time.   Solution: Aging  (b).  IMG_20210502_212113  IMG_20210502_212119 |
| 1. A.Define and differentiate between preemptive & non-preemptive scheduling. [2] 2. Consider process arrival as given below where N = right most significant digit of your Roll No.( ex:- for Roll No. 18085**4**, N=**4**):  |  |  |  |  | | --- | --- | --- | --- | | Process | CPU Burst Time(ms) | Arrival Time | Priority | | A | 4 | 0 | 2 | | B | 3 | 3 | 3 | | C | 6 | 4 | 1 | | D | 5 | 6 | N | | E | 1 | N | 4 |   Calculate the following for *priority (non preemptive)* and *round robin* (time quantum = 2 ms) CPU scheduling algorithm: [10]  (a)Average waiting time  (b)Turnaround time for each process  (c)Order of completion  (*hints:-higher digits indicate higher priority*)  Solution:  Solution for Non-Preemptive Priority  Example:- Roll No. 18085**4**, N=**4**   |  |  |  |  | | --- | --- | --- | --- | | Process | CPU Burst Time (ms) | Arrival Time | Priority | | A | 4 | 0 | 2 | | B | 3 | 3 | 3 | | C | 6 | 4 | 1 | | D | 5 | 6 | N =4 | | E | 1 | N = 4 | 4 |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | A | E | B | D | C |   0 4 5 8 13 19   1. Each process waiting time   A = 0 – 0 = 0  B = 5 – 3 = 2  C = 13 – 4 = 9  D = 8 – 6 = 2  E = 4 – 4 = 0  Avg. Wait Time = 13/5 = 2.6  (b)Turnaround time for each process  A = 4 + 0 = 4  B = 3 + 2 = 5  C = 6 + 9 =15  D = 5 + 2 = 7  E = 1 + 0 = 1  (c) Order of completion: - A-E-B-D-C  Solution for round robin (time quantum = 2 ms) CPU scheduling algorithm:   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Process | CPU Burst Time (ms) | Arrival Time | Wait Time | TAT for each Process | Execution time | | A | ~~4~~  ~~2~~ 0 | ~~0~~ ~~2~~ 4 | (0-0) + (2-2) = 0 | 4 | 0 + 2 + 2 +2+2+1+1+2+2+2+2 +1 =19 | | B | ~~3~~ ~~1~~ 0 | ~~3~~ ~~6~~ 10 | (4-3) + (9-6) = 4 | 7 | | C | ~~6~~ ~~4~~  ~~2~~ 0 | ~~4~~  ~~8~~ ~~14~~ 18 | (6-4) + (12-8) +  (16-14) = 8 | 14 | | D | ~~5~~ ~~3~~ 1 | ~~6~~ ~~12~~ 16 | (10-6) + (14-12) + (18-16) = 8 | 13 | | E | ~~1~~ 0 | N = ~~4~~  9 | 8 – 4 = 4 | 5 |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | A | A | B | C | E | B | D | C | D | C | D |   0 2 4 6 8 9 10 12 14 16 18 19  (a)Average waiting time = 24/5= 4.8  (b)Turnaround time for each process (Available in the table)  (c)Order of completion = A – E – B – C - D |
| **Q.No:9** | (a)What is “starvation" of process and how to prevent it? What do you mean by cascading termination? (6+6)  (b)Find the total head movement for FCFS scheduling with following disk queue requests on I/O blocks: 98, 183, 37, 122, 14, 124, 65, 67. Head starts at N (where N= your Roll No. MODULUS 100) .  Solution:   1. Starvation is the problem that arises when a process with higher priority keeps executing and low priority processes get blocked for indefinite time. In a heavily loaded computer system, a continuous execution of higher-priority processes can prevent a low-priority process from ever getting the CPU. The phenomenon of starvation can be prevented by ‘Aging’.   \*\* Students may include the definition of aging to explain the concept.  When a parent process is terminating , then all of its children process is also **terminated**. This phenomena is known as "**Cascading Termination**" and is normally initiated by the operating system.  b) The disk queue requests are: **98, 183, 37, 122, 14, 124, 65, 67.** Let us assume that the head is at block = 53.  Then A/ FCFS scheduling Total disk movement = 640.  |98-53| + |183-98| + |37-183| + |122-37| + |14-122|+ |124-14| + |65-124| + |67-65|. The students may look to draw the diagram and explain for better understanding. |
| (a)In terms of the variable pid, differentiate between a child process and a parent process? Draw the Queuing diagram representation of process scheduling for all types of schedulers. (6+6)  (b)Find the total head movement for SSTF scheduling with following disk queue requests on I/O blocks: 98, 183, 37, 122, 14, 124, 65, 67. Head starts at Head starts at N (where N= your Roll No. MODULUS 100) .  Solution:  a) The students may explain with full code. That will be entitle them to better understanding and presentation. In brief:  If pid < 0,then no child process is created.  If pid = 0, then the child process will execute its own code segment separate from the parent process.  If pid > 0, then this means the child process is successfully created, and the parent process will execute its own segment of code.  G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\Queueing-diagram.png  G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\midterm-quing.png  b) The total head movement required given that the head =53 is 236. The policy is to go to that request or I/O block that is nearest to the current request being processed. |
| 1. What are the criteria for a good CPU scheduling algorithm? State the differences between Multi level queue scheduling(MQS) and Multi level feed back queue scheduling(MFQS)? Which of MQS or MFQS can lead towards starvation of processes? (6+6)   Find the total head movement for SCAN and C-SCAN scheduling with following disk queue requests on I/O blocks: 0, 14, 37, 53, 65, 67, 98, 122, 124, 183. Head starts at N (where N= your Roll No. MODULUS 100) . Which algorithm is more preferable and why?  Solution:  **a) Criteria for good CPU scheduling algo:**  **CPU utilization** – keep the CPU as busy as possible  • **Throughput** – # of processes that complete their execution per time unit  • **Turnaround time** – amount of time to execute a particular process  • **Waiting time** – amount of time a process has been waiting in the ready queue  • **Response time** – amount of time it takes from when a request was submitted until the first response is produced, not output (for timesharing environment.  MQS: Partitions the ready queue into multiple queues. Each queue contains processes based on properties such as memory size, process priority or process type. Each Queue may have unique scheduling algorithms.  MFQS: It allows the processes to move between the queues. The idea is to separate the processes based on the CPU bursts. If the process uses too much of CPU time, it will be moved to the lower priority process.  MFQS may lead towards starvation due to indefinite waiting of a process in the lower priority queue.  b) Assuming that the disk arm is moving towards 0 and the initial head position is at 53.  G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\scan.png  No. of head movements=236.  With C-SCAN , assuming that head is at 53 and the movement is towards the 0th block, then the equivalent head movement would be : 187. However students may assume a certain direction of traversal/ processing request. Out of the two, the one with lesser head movement is more preferable. |
| **Q.No:10** | Consider a paging memory management system, where each entry of the page table consists of frame address, valid bit(1 bit), present bit(1 bit), protection bit(2 bits), modified bit(1 bit). The virtual address is 18bits, page size is 512 bytes, and page table is exactly fit into one page. The program is allocated with 3 page frames. The program generates the following 16 page numbers (which are part of the virtual address produced by ): 0, 17, 18, M, 20, 2, 20, M, 17, 32, 0, 2, 2\*M, 0, 16, M (Note: The page numbers are in decimal and M=sum of digits of your Roll No. MODULUS 10). **(6+6)**     1. Calculate the number of page faults generated for the above page request, assuming a Optimal page replacement algorithm. 2. Find the maximum size of the program for single level paging.   Solution:  (a)  1  (b)  1 |
| Consider a paging memory management system, where each entry of the page table consists of frame address, valid bit(1 bit), present bit(1 bit), protection bit(3 bits), modified bit(1 bit). The virtual address is 18bits, page size is 512 bytes, and page table is exactly fit into one page. The program is allocated with 3 page frames. The program generates the following 16 page numbers (which are part of the virtual address produced by ): 0, 17, 18, M+2, 20, 2, 20, M, 31, 32, 0, 18, N, 0, 17, M (Note: The page numbers are in decimal and M=sum of digits of your Roll No. MODULUS 10). **(6+6)**   1. Calculate the number of page faults generated for the above page request, assuming a Optimal page replacement algorithm. 2. Find the maximum size pf the program for single level paging.   Solution:   1. Page frames = 3   Ref. Str : 0, 17, 18, M+2, 20, 2, 20, M, 31, 32, 0, 18, N, 0, 17, M  N = not defined . Based on the assumption of the student the answer will be evaluated.  Consider    b.  Page size= 512 bytes = 29  VA = 18 bits  valid bit =1 bit  present bit = 1 bit  protection bit= 3 bits  modified bit = 1 bit  Frame size = Page size = 512 B = 512 X 8 bits   |  |  | | --- | --- | | 9 | 9 |   Page Table entry size (PTES)= 1+ 1 + 3 + 1+ 512 X 8 bits = 6+ 4096 = 4112 = 512. 875 bytes  V Address Space (VAS) = 218 bytes  No of pages (N) = VAS / PS = 218 / 29 = 29  Page Table Size = N X PTES = 512 X 512.75 B = 256.375 KB  So, Maximum size of program for the single level of paging = 512 X 512 B = 256 KB |
| Consider a paging memory management system, where each entry of the page table consists of frame address, valid bit(1 bit), present bit(1 bit), protection bit(2 bits), modified bit(1 bit). The virtual address is 18bits, page size is 512 bytes, and page table is exactly fit into one page. The program is allocated with 3 page frames. The program generates the following 16 page numbers (which are part of the virtual address produced by ): 0, 17, 18, M, 20, 2, 20, M+4, 17, 32, 0, 2, M+2, 0, 16, M (Note: The page numbers are in decimal and M=sum of digits of your Roll No. MODULUS 10). **(6+6)**     1. Calculate the number of page faults generated for the above page request, assuming a Optimal page replacement algorithm.   II. Find the maximum size of the program for single level paging.  Solution:  2 |
| **Q.No:11** | Find a safe sequence(if any) for the following resource allocation table using deadlock detection algorithm.  Process Alloc Req Avail  P0 0 1 0 2 2 2 4 3 3  P1 1 1 1 1 2 2  P2 1 0 1 3 2 2  P3 2 1 2 4 2 3  P4 0 0 0 3 2 3  Does the system has deadlock, if so then which are the deadlocked processes? Also find any unsafe sequence and provide the value of ‘Work’ vector for it. (6+2+4)  Solution:  **G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\14494161101325_14494170811632.jpg**  **G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\14494161101325_14527855728942.jpg** |
| Find a safe sequence(if any) for the following resource allocation table using Banker’s algorithm.  Process Alloc Max Avail  P0 0 1 0 2 2 2 4 3 3  P1 1 1 1 1 2 2  P2 1 0 1 3 2 2  P3 2 1 2 4 2 3  P4 0 0 0 3 2 3  Also find any unsafe sequence and provide the value of ‘Work’ vector for it. processes? (6+4+2)  Solution:  **G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\14494161101325_14527855728942.jpg** |
| Convert the following Resource Allocation Graph (RAG) to Wait for Graph (WFG)  R1  P2  P1  P3  R4  R3  Identify the cycle in the corresponding WFG. List the processes which are a part of deadlock.  Solution:  G:\Jobs\KIIT-20-21\KIIT-Dec-Apr-2021\OS-CSE-G2\Sem-End-OS\14494161101325_14554917147944.jpg |