**MCQs**

1. What is process scheduling in an operating system?
2. Allocating memory to processes
3. Managing the execution order of processes
4. Allocating CPU time to processes
5. Managing file access by processes
6. Which scheduling algorithm aims to minimize the average waiting time for processes?
7. First-Come-First-Serve (FCFS)
8. Shortest Job Next (SJN)
9. Round Robin (RR)
10. Priority Scheduling
11. In a preemptive scheduling algorithm, a higher priority process can interrupt the execution of a lower priority process. Which algorithm is an example of preemptive scheduling?
12. FCFS
13. Shortest Job Next
14. Priority Scheduling
15. Round Robin
16. The time taken by a process from the time of submission to the time of completion is called:
17. Turnaround time
18. Waiting time
19. Execution time
20. Response time
21. What is the main drawback of the First-Come-First-Serve (FCFS) scheduling algorithm?
22. It may lead to starvation
23. It doesn't utilize CPU time efficiently
24. It has high overhead
25. It requires complex prioritization rules
26. The Shortest Job Next (SJN) scheduling algorithm selects the process with the:
27. Longest execution time
28. Shortest execution time
29. Highest priority
30. Lowest priority
31. If a Round Robin scheduling algorithm is used with a time quantum of 20 milliseconds, and each process gets the same amount of CPU time, how long will it take to complete the first process if it requires 60 milliseconds of CPU time?
32. 20 milliseconds
33. 40 milliseconds
34. 60 milliseconds
35. 80 milliseconds
36. Which scheduling algorithm allows processes with the same priority to execute in a first-come-first-serve order?
37. FCFS
38. SJN
39. Priority Scheduling
40. Round Robin
41. In Priority Scheduling, how are the priorities of processes usually assigned?
42. In ascending order of arrival time
43. In descending order of burst time
44. In descending order of priority values
45. In ascending order of process IDs
46. A process that has completed execution and has exited is said to be in the:
47. Ready state
48. Running state
49. Blocked state
50. Terminated state
51. What is a race condition in computer science and programming?
52. A competition between software developers to write code faster.
53. A situation where multiple threads or processes access shared data concurrently, leading to unexpected results.
54. A condition where code is executed sequentially, ensuring data consistency.
55. A situation where only one thread or process can access shared data.
56. Which of the following is a common consequence of a race condition?
57. Predictable and consistent program behavior.
58. Deadlock.
59. Data corruption or incorrect results.
60. Improved program performance.
61. To mitigate race conditions, which synchronization technique is commonly used to ensure exclusive access to shared resources?
62. Multithreading
63. Lock-free programming
64. Mutex (Mutual Exclusion)
65. Thread pooling
66. Which of the following is NOT a potential solution to preventing race conditions?
67. Using semaphores or mutexes.
68. Employing multi-core processors.
69. Implementing thread-safe data structures.
70. Applying proper synchronization techniques.
71. In a multi-threaded program, when two threads simultaneously access and modify a shared variable, what may be needed to prevent a race condition?
72. A faster processor.
73. Synchronization mechanisms like locks or semaphores.
74. A larger stack size for each thread.
75. Reordering the code execution sequence.