**Operating System Functionalities:**

1. Explain dual mode operations in OS through classes of interrupts. Illustrate the purpose of syscall, timer, illegal memory access, and illegal instruction handlers.
2. Analyze how operating systems manage and control processes, including the role of Process Control Blocks (PCBs). Discuss the significance and purpose of the Process API in this context.

**Types of Operating Systems**

1. Conduct a case study of various operating systems including DOS, Windows, UNIX, Linux, Mac OS, Android, and iOS. Analyze their histories, architectures, features, and their impacts on the computing world.

**Specialized Operating System Functions**

1. Demonstrate the steps taken by the shell to execute a command using fork(), exit(), wait(), and execve(). Explain how parent and child processes share open files before the child process is created and describe how file offsets are modified by read(), write(), and lseek().
2. Analyze the Limited Direct Execution protocol, particularly focusing on how it handles context switches using timer interrupts to regain control. Evaluate the necessity of the process table in a timesharing system and discuss its role in managing processes.

**CPU Scheduling**

1. Assume you have the following jobs to execute with one processor, with the jobs arriving in the order listed here: Construct a Gantt chart illustrating the execution of these processes. Calculate the waiting time, turnaround time, and completion time of each process for each of the given scheduling algorithms: • Shortest Time-to-Completion First (STCF) • Round Robin with a time quantum of 2. Process: P1, P2, P3, P4 Arrival Time: 0, 1, 2, 3 Burst Time: 5, 4, 2, 1
2. Assume you have the following jobs to execute with one processor, with the jobs arriving in the order listed here: Construct a Gantt chart illustrating the execution of these processes. Determine the waiting time, turnaround time, and completion time of each process for each of the given scheduling algorithms: • Shortest Time-to-Completion First (STCF) • Round Robin with a time quantum of 2. Process: P1, P2, P3, P4 Burst Time: 5, 4, 2, 1
3. Assume you have the following jobs to execute with one processor, with the jobs arriving in the order listed here: • Process: P1, P2, P3, P4 • Arrival Time: 0, 1, 2, 3 • Burst Time: 5, 4, 2, 1. Construct a Gantt chart illustrating the execution of these processes. Determine the waiting time, completion time, and turnaround time of each process for each of the given scheduling algorithms: a) FCFS b) SJF
4. Demonstrate single-queue and multi-queue multiprocessor scheduling, and identify the issues of cache affinity and load imbalance with examples.
5. Analyze the scheduling of four processes P1, P2, P3, and P4 using the Shortest Remaining Time First and Priority Scheduling algorithm, where: CPU Burst Times are 6, 8, 7, and 3 ms respectively. Priority Orders are 4, 1, 3, and 2 respectively (with lower numbers indicating higher priority). Arrival Time for all processes is zero. Create a Gantt chart illustrating the execution of these processes using Priority Scheduling. Calculate the Average Waiting Time (AWT) and Average Turnaround Time (ATAT) by using the above Scheduling algorithms.
6. Analyze the Shortest Job First (SJF) scheduling algorithm, including its advantages and disadvantages. Calculate and compare the average waiting times using both FCFS and SJF algorithms for a set of four processes with CPU burst times of 5, 10, 8, and 3 ms.
7. Apply the FCFS and Shortest Remaining Time First (SRTF) scheduling algorithms to a set of processes P1 to P5 with CPU burst times of 3, 6, 4, 5, and 2 units and arrival times of 0, 2, 4, 6, and 8 units, respectively. Identify the Average Wait Time and Turnaround Time for each algorithm.
8. Illustrate the Round-Robin scheduling technique using a set of three processes P1, P2, and P3 with CPU burst times of 24, 3, and 3 units, respectively. Calculate the average waiting time with a time quantum of 4 ms.
9. Analyze the Round Robin scheduling algorithm with a time quantum of 2 ms for four processes with burst times of 5, 4, 2, and 1 ms and arrival times of 0, 1, 2, and 4 ms. Calculate the average waiting time and turnaround time and discuss the effect of the time quantum on performance.

**Multilevel Feedback Queues**

1. Evaluate the concept of multilevel feedback queues in scheduling. Analyze how this approach balances responsiveness and fairness among processes.
2. Examine the core principles of Multilevel Feedback Queue Scheduling. Analyze its strategy for managing processes with varying priorities, and discuss the advantages and constraints of this scheduling approach in a multi-user environment. Provide a practical example to illustrate how it operates.

**Lottery Scheduling**

1. Demonstrate the steps taken by the shell to execute a command using fork(), exit(), wait(), and execve(). Explain how parent and child processes share open files before the child process is created and describe how file offsets are modified by read(), write(), and lseek().
2. Illustrate the concept of lottery scheduling to a scenario with two processes, where one has 75% of the tickets and the other has 25%. Explain how this scheduling method would manage their execution.
3. Analyze the Lottery Scheduling algorithm, including how tickets are assigned to processes, and evaluate the benefits and challenges of implementing this algorithm in an operating system.

**Multiprocessor Scheduling**

1. Demonstrate single-queue and multi-queue multiprocessor scheduling, and identify the issues of cache affinity and load imbalance with examples.