Case Study 1: VIRTUAL MEMORY

Implementing virtual memory concepts

Virtual memory is a very interesting subject since it has so many different aspects: page faults, managing the backing store, page replacement, frame allocation, thrashing, and page size. A simulation is probably the easiest way to allow the programmer to program several of the pagereplacement algorithms and see how they really work. If an interactive graphics display can be used to display the simulation as it works, the programmer may be better able to understand how paging works. Page replacement algorithms are used to optimize the memory usage.

Write a Java program that implements the FIFO and LRU page replacement algorithms

Case Study 2: CPU Scheduling

Implementing CPU scheduling in multiprogrammed operating systems. By switching the CPU among processes, the operating system can make the computer more productive. FCFS, SJF, Round-Robin, Priority scheduling algorithms are the mostly used scheduling algorithms. The idea of resource allocation and scheduling, Gantt charts, simulations, and play acting are valuable ways to get the ideas across. Show which of the ideas should be used in some situations like waiting in line at a post office, a waiter time sharing between customers, even classes being an interleaved Round-Robin scheduling of professors.
Consider the situation in which following set of processes, with the length of the CPU-burst time given in milliseconds:

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at the same time
Implement a perfect scheduling algorithm so that the waiting time for all the processes is minimum.

Case Study 3: Deadlock

Implementing resource-allocation policy
Requests and releases for resources are allowed at any time. If a request for resources cannot be satisfied because the resources are not available, then we check any processes that are blocked, waiting for resources. If they have the desired resources, then these resources are taken away from them and are given to the requesting process. The vector of resources for which the waiting process is waiting is increased to include the resources that were taken away.
For example, consider a system with three resource types and the vector Available initialized to (4, 2, 2). If process P0 asks for (2, 2, 1), it gets them. If P1 asks for (1, 0, 1), it gets them. Then, if P0 asks for (0, 0, 1), it is blocked (resource not available). If P2 now asks for (2, 0, 0), it gets the available one (1, 0, 0) and one that was allocated to P0 (since P0 is blocked). P0's Allocation vector goes down to (1, 2, 1), and its Need vector goes up to (1, 0, 1).
What could be the situations which can lead to deadlock or indefinite blocking in the above problem statement?

Case Study 4: COMPUTER-SYSTEM STRUCTURES

Implementing Prefetching is a method of overlapping the I/O of a job with that job's own computation.
The idea is simple. After a read operation completes and the job is about to start operating on the data, the input device is instructed to begin the next read immediately. The CPU and input device are then both busy. With luck, by the time the job is ready for the next data item, the input device will have finished reading that data item. The CPU can then begin processing the newly read data, while the input device starts to read the following data.
A similar idea can be used for output. In this case, the job creates data that are put into a buffer until an output device can accept them.
Compare the prefetching scheme with the spooling scheme, where the CPU overlaps the input of one job with the computation and output of other jobs.