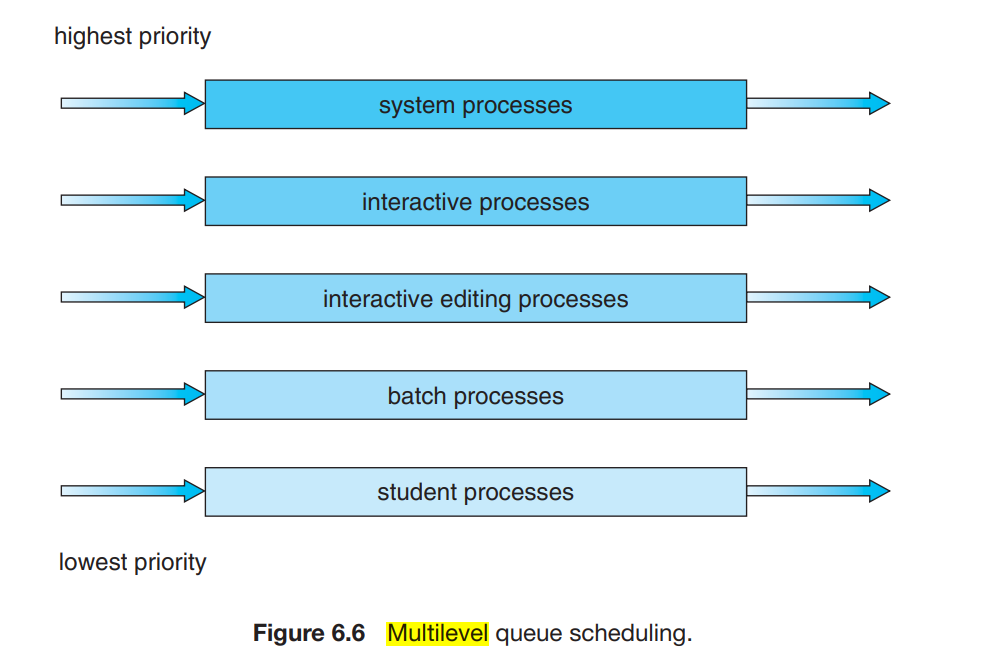
**Describe the issues associated with MLFQ and also provide the solutions for the same.**

**Problems in the above implementation** –

A process in the lower priority queue can suffer from starvation due to some short processes taking all the CPU time.

**Solution** –

A simple solution can be to boost the priority of all the processes after regular intervals and place them all in the highest priority queue.



A multilevel queue scheduling algorithm partitions the ready queue into

several separate queues (Figure 6.6). The processes are permanently assigned to

one queue, generally based on some property of the process, such as memory

size, process priority, or process type. Each queue has its own scheduling

algorithm

In addition, there must be scheduling among the queues, which is commonly implemented as fixed-priority preemptive scheduling. For example, the foreground queue may have absolute priority over the background queue.

Let’s look at an example of a multilevel queue scheduling algorithm with

five queues, listed below in order of priority:

1. System processes

2. Interactive processes

3. Interactive editing processes

4. Batch processes

5. Student processes

Each queue has absolute priority over lower-priority queues. No process in the batch queue, for example, could run unless the queues for system processes, interactive processes, and interactive editing processes were all empty. If an interactive editing process entered the ready queue while a batch process was running, the batch process would be preempted

C program to create a process using fork() system call

