1. The OS quantum time is the is the amount of CPU time allocated to a process. If the quantum expires before the process voluntarily relinquishes the CPU, it is placed in the Level 2 queue in a Multilevel Feedback Queses.
2. If the quantum is large enough, then RR reduces to the FCFS algorithm;
3. If it is very small, then each process gets 1/nth of the processor time and share the CPU equally.
4. You would know you have the right quantum size when you can maximise throughput and achieve balance between response time and utilisation.
5. A switch between 2 threads in the same process is easier because It takes less time to terminate an existing thread, or to switch between two threads within the same process
6. if the kernel is single threaded, then any user level thread performing a blocking system call will cause the entire process to block, even if other threads are available to run within the application. If a process is running, and turned into a blocked state, it passes control to another process. However According to the data structure maintained by the threads library, threads associated with the process is still running.
7. A combined ULT/KLT approach in a multithreaded program would run faster because Multiple threads of the same process can be scheduled simultaneously on multiple processors. If one thread becomes blocked, the kernel can schedule another thread from the same process. This undoes the cons of ULTs of the kernel assigning one process to only one processor at a time and multiple threads unable to run on multiprocessors.
8. (1)Save the context of the processor (eg program counter and other registers)

(2)Update the PCB of the process currently running

(3)move the PCB of this process to the appropriate queue (eg. blocked, suspended)

(4)Select another process for execution

(5)Update the PCB of the selected process

(6)Update memory management data structures

(7)Restore the context of the processor to that which existed at the time the selected process was last switched out of the running state.

1. b. The process issued an I/O request
2. ULT:

* Kernel mode privileges not required. Therefore the process does not switch to kernel mode to do thread management (saving two context switches).
* Scheduling can be application specific.
* ULTs can run on any OS.

KLT:

* Because kernel has full knowledge of all threads, Scheduler may decide to give more time to a process having large number of threads than process having small number of threads.
* Kernel-level threads are especially good for applications that frequently block.