Linux programming assignment – 1

Note: one way to achieve uniprocessor mode of execution for our active applications/processes is to use taskset utility – refer to manual page of taskset utility for more details !!! also, refer to our lecture notes !!!

1. using while1.c provided, do the following:
   - compile and load 2 instances, in the background – check with ps and top commands, as discussed, in lectures !!!
   - next, compile and load 2 more instances, in the background – observe, using ps and top commands, as discussed, in lectures !!!
   - next, compile and load 2 more instances, in the background – observe, using ps and top commands

Note1 - you need to generate w1, using `gcc while1.c -o w1`

Note 2 - use `taskset 0x00000001 ./w1&` or `taskset 0x00000002 ./w1&`, or another cpu mask to launch your applications,as per requirements – this will launch your applications on a specific processor / scheduler instance

What do you observe, with respect to usage of cpu cycles on each processor /scheduler instances ??Reason your observations !!!

2. now, repeat the above problem, by changing the nice values of each instance, using renice command, as per lecture notes/pdfs – use renice to modify the values in +ve range only, like +5 | +10 | +15 - observe using ps and top commands – what are your conclusions ???

Note : with administrative privileges, you can change the nice priority values to -ve values, which will provide very large time-share values to processes !!! you can assign -5 | -10 | -10 | -20 to different processes on a given processor/scheduler instance ??

Note : refer to scheduling_2_class1n.pdf, for usage of renice command - also, refer to manual pages of renice !!!

3. now, using chrt utility, can you modify scheduling policy of active applications/ processes, in problems 1) to FIFO or RR, with appropriate real-time priorities – use the following policy and priority assignments ???
   a) assign FIFO with equal priority to a few processes and assign TS policy to a few processes – you need to assign these processes to a specific processor / scheduler instance
   b) assign RR with equal priority to a few processes and test – once again, you need to assign these processes to a specific processor/scheduler instance

3. c) assign FIFO, with unequal priorities to a few processes and test – once again, assign these processes to a specific processor / scheduler

   d) assign FIFO with equal priority to a few processes and assign TS policy to a few processes – you need to assign these processes to a specific processor / scheduler instance – in addition, add sched_yield() to active applications/processes, which are assigned FIFO policy – do not add sched_yield() to active/applications/processes assigned, with TS policy

Note : verify the behaviour of the above processes,
using ps and top commands.

Note : in the above scenarios, the system may stop responding, due to
starvation of normal application processes and system processes.
Just be warned ???