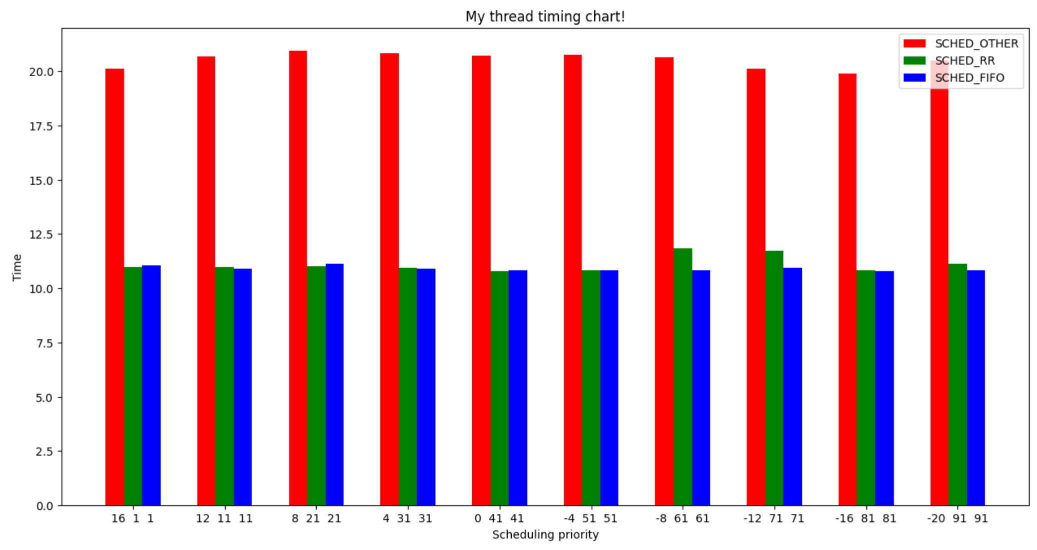
# Linux pthreads and their scheduling

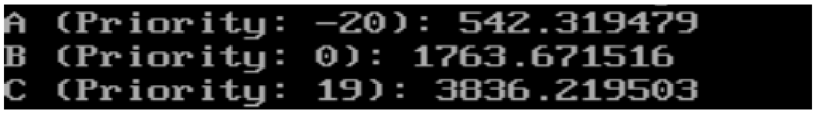
1. Thread A executed with SCHED_OTHER policy has nice value ranging from 16 to -20 in steps of 4, it takes highest time compared to other threads with SCHED_RR and SCHED_FIFO policy
2. Thread B and C are respectively with SCHED_RR and SCHED_FIFO policy with their priority changed from 1 to 91 in steps of 10, they take almost similar time and their time is approx. ½ of time take by SCHED_OTHER policy
3. All these threads are executed with root permission so as to change their nice value or priority



4. API used to set scheduling policy and priority for Thread B and C with SCHED_RR and SCHED_FIFO policy
    a. pthread_setschedparam()
5. The above API is also used to set scheduling policy for SCHED_OTHER and following API is used to set the nice value
    a. nice()
6. Time is measured using clock_gettime() with clockid_t as CLOCK_REALTIME to compare wall clock time for all threads

## Process scheduling

1. Three process are created here with their priority as -20, 0 and 19 with -20 being highest priority. These processes compiled the custom kernel using “make -j$(nproc)” where $(nproc) = 2 on my m/c
2. The time taken by different processes is as follows. It can be seen that process with priority -20 (highest) complies in ~542s whereas one with least priority takes ~3836s



```
A (Priority: -20): 542.319479
B (Priority: 0): 1763.671516
C (Priority: 19): 3836.219503
```

3. The processes were created with fork() and each process executed a bash script which internally does ‘cd’ to linux code area and then does “make -j$(nproc)”, execl() system call was used to execute the shell script in each of these process.
4. The process priority was changed using “setpriority()” call while running with root permission.
5. waitpid() non-blocking call was done to poll when a child process has finished to record its end time
6. Time here was measured using clock_gettime() with clockid_t as CLOCK_PROCESS_CPUTIME_ID