Exercise Problems on Scheduler system calls

1.Write a C program that creates two threads using the pthread library.
One thread should set its scheduling policy to SCHED_FIFO with a priority of 1,
and the other thread should set its scheduling policy to SCHED_RR with a priority of 2.
Have both threads perform some CPU-bound tasks, and observe how the
different scheduling policies affect their execution.

2.Create a C program that uses the sched_getaffinity() and sched_setaffinity() system
calls to bind a process to a specific CPU core. Measure the performance of your program
on different CPU cores and analyze the results. How does CPU affinity affect the
program's execution time?

3.Implement a multi-threaded program where each thread has a different priority using
the sched_setscheduler() and sched_setparam() functions. Have each thread perform a
specific task and measure the time it takes for each thread to complete.
Analyze the results in terms of scheduling priority and execution times.

4.Write a C program that uses the nice() function to adjust the niceness value of a process.
Create multiple processes with different niceness values and observe how this
affects their CPU time allocation. Explain the impact of niceness on process scheduling.

5.Implement a C program that uses the sched_yield() function to voluntarily
yield the CPU by one of its threads. Create multiple threads, including one that
frequently calls sched_yield(). Measure the impact of yielding on the overall
system's CPU utilization.