Goal:

To understand, use and analyze performance of synchronization mechanisms in Linux kernel. In this assignment, we will use various lock implementations offered by kernel and add custom implementation for a read-write lock. Two parts of the assignment are explained below.

PART 1

Get a handle of the sample code (attached), go through the README and understand the code organization. Follow the steps to compile the provided module and execute user-space micro benchmark (syncbench) to understand its usage and output formats. The sample code and template implements the in-built kernel spinlock synchronization mechanism. As part of the assignment you are required to extend the kernel module to implement the following,

(i) use kernel API to use read-write locks, sequential locks and RCU

(ii) design a custom read-write lock and use it just like using a kernel provided locking scheme. Verify both the above implementations for correctness.

PART 2

Perform an empirical comparison of all locking schemes (implemented in PART 1). The comparison should focus on overheads considering read-write ratio as a parameter for comparison. Following guiding principles may be helpful in empirical analysis,

(i) Clear explanation of experimental setup, workload and parameters etc.

(ii) Explanation of results, verifying statistical sanctity (run-time, avg., deviation etc.)

(iii) Result insights

Extra credit

Design a new locking scheme to improve performance beyond the best locking scheme found in the empirical analysis (PART 2).

Submission format

Submissions should be one single archive (named asg2-rollno.tgz) with the following structure
when expanded.

```
asg2-rollno
    |
    |---------src        source code, Makefile and an implementation notes text file
    |----------run
               |------ correctness.txt                Output of tests showing correctness of your
               |                                       implementation
               |----- unit-test.sh (or .py etc.)      when executed, it will execute all the
               |                                       correctness tests reported in correctness.txt
               |----analysis-test.sh (or .py etc.)    when executed, it will execute all tests which
               |                                       generates data used to write your empirical
               |                                       analysis report inside doc folder
   |----------doc
                  |------- report.pdf
                  |------- plots and figures
```

The report should be a document which enables the reader---to understand your experiments and the findings, to re-run the experiments if she desires so.

"Do not defeat your purpose of being here by cheating"
IITK anti-cheating policy: https://www.cse.iitk.ac.in/pages/AntiCheatingPolicy.html