CS451: **Introduction to Parallel and Distributed Computing**
Spring 2026

Assignment 2: Refresher on UNIX, C, and its toolchain

Setup:

1. Create a FABRIC slice that contains a single node provisioned with Ubuntu 22.04.5.
2. On that node, install GCC, Valgrind, and gprof.
3. Compile libcompart (Shohola version) (http://www.cs.iit.edu/~nsultana1/files/libcompart_shohola.tgz)
4. Compile and run the "hello_compartment" example that comes with libcompart.

Baseline:

5. Compile and run the code that computes Life.
6. Run the Life program for 100 cycles and time its execution by using `command time` in bash.
7. Using Valgrind, check the Life binary for memory leaks.
8. Using gprof, profile the Life binary.

Development:

9. Building on "hello_compartment", partition the code that computes Life by putting the `step()` function in a separate compartment. Remember to serialize the world's state when moving between compartments.
10. Run the (updated) Life program for 100 cycles and time its execution.
11. Using Valgrind, check the (partitioned) Life binary for memory leaks.
12. Using gprof, profile the (partitioned) Life binary.

What to submit: A .tgz file containing a set of text files:

- A file called `partitioned_life.c` containing the code of your partitioned Life program from step 8, and any other `.c` and `.h` files that accompany the partitioned program.

- A set of files named "`a.out`", "`a.err`", "`b.out`", etc, each containing the contents specified below:

a. The (compiler's) output from step 4.
b. The (compiler's) output from step 5.
c. The output from running the Life program (from step 6).
d. The output from step 7.
e. The output from step 8.
f. The (compiler's) output from step 9.
g. The output from running your program (from step 10).
h. The output from step 11. **(Bonus: +10% if there are no memory leaks)**

i. The output from step 12.
j. **Bonus: (+10%)** Repeat step 6 for 5 times and compute the average and standard deviation of the total invocation time.

Where files ending in `.out` should contain the standard output, and files ending in `.err` should contain the error output.