COMP 2401 -- Tutorial #10
Processes and Threads

Learning Objectives

After this tutorial, you will be able to:

- use fork to create child processes
- use threads to distribute computationally intensive operations

Tutorial - Forking

This week's tutorial will modify the `send` / `handle` program from last week so that it becomes a parent / child process with the parent sending signals to the child.

1. Download the file `T10.tgz` from the tutorial page in *cuLearn*. Extract, and read through the tutorial files.

2. Write a program in `t10-fork.c` which creates a new process in `main()` using the `fork` system call. Using the return value from `fork()`, print out whether the process is the child or the parent.

3. Add signal handlers to *only* the child process so that it prints to `stdout` when it receives `SIGUSR1`, `SIGUSR2`, or `SIGINT`. The child should then wait until receiving `SIGINT` before terminating.

   Hint: the child's code should look very similar to the code for `handle` *from Tutorial 9.*

4. Add code for (only) the parent process so that it prompts the user to choose which signal to send to the child, then send it. The process should terminate once it has sent a `SIGINT` to the child process.

Tutorial – Threading

5. Speed up `t10-threads` by using multiple threads to determine if the ten numbers are prime. You should not make any changes to the `prime()` function itself.