Lab: Process API and Thread API

Outcomes

- Create and Control Processes
- fork(), wait(), and exec() signal calls
- Threads Creation and Completion
- Locks and Condition Variables

Online Resources

- **Operating Systems: Three Easy Pieces:** https://pages.cs.wisc.edu/~remzi/OSTEP/
- **Homework page:** https://pages.cs.wisc.edu/~remzi/OSTEP/Homework/homework.html
- **Process API Reading**: https://pages.cs.wisc.edu/~remzi/OSTEP/cpu-api.pdf
- **Threads API Reading**: https://pages.cs.wisc.edu/~remzi/OSTEP/threads-api.pdf

---

Tasks

1. Do coding questions 1,3,5,6 and 8 on page 14 of the Process API Reading.

Save screenshots of your output and answers to the questions in a document. Save your code.

2. Read through the Threads API Reading

(You can leave the section below until next week but it's a good idea to compile before next week's lab in case you need help)

3. Download and compile the code in the Threads API Reading.

   This is explained in Section 27.5 Compiling and Running of the reading, on page 9.

   Here is the link to the code:

   http://pages.cs.wisc.edu/~remzi/OSTEP/Homework/HW-Threads-RealAPI.tgz

   You can individually compile the code e.g.

   ```
   gcc -o main-race main-race.c -Wall -pthread
   ```

   But much faster initially to do the following:

   (Make sure you are still within threads-api folder in the container first!!!)

   ```
   mkdir build && cd build
   cmake ..
   cd ..
   make
   ```

   Read and follow instructions in the README file and check that you can run the commands from the Threads API homework as follows (note this is slightly different than what the book tells you!):

   ```
   valgrind --tool=helgrind ./main-race
   ```