ELE709 - Real-Time Computer Control Systems
Lab 3 - POSIX Threads and Concurrent Programming (Week 1)

Name: Toni Pano #500822828 Section 3

Timing information should be obtained by running the required programs for the Exercises on a workstation in ENG413.

1. **Exercise 3.2**

   (a) Record the required execution time in Table A.1 below.

   (b) Should these execution time be similar to those obtained in Lab 2? Explain.

   The execution time for Exercise 3.2 should be similar to those obtained in Lab 2 because they are running the same type and volume of math operations. In both cases, the clock time is measured directly before and after a for loop with a specific math operation executes. This would make the elapsed time the same in both cases. In practice, the code in Lab 2 runs slightly slower than the code in Exercise 3.2 for all 4 math operations.

2. **Exercise 3.3**

   (a) Record the required execution time in Table A.1 below.

   (b) Compare the results obtained for Exercises 3.2 and 3.3. Are the results similar. Explain why.

   The execution time for Exercise 3.2 is slightly faster than the execution time for Exercise 3.3. This is because Exercise 3.3 makes all 4 of its threads run concurrently. If a thread is temporarily blocked by another thread, the elapsed time measured for the first thread will include a part of the execution time for the other threads' instructions. This would make the execution time for each thread be longer than what it would be if no threads switched state during execution, like in Exercise 3.2. Because Exercise 3.3 makes 4 threads run concurrently, they are likely to be blocked periodically so that CPU access can be shared among all threads.

Table A.1: Execution Time per Iteration (in nanoseconds)

	Exercise 3.2					Exercise 3.3			
	+	−	×	÷		+	−	×	÷
1	1.950253	1.828700	1.807434	3.603248	1	2.036742	2.048937	2.028989	3.827331
2	1.947160	1.847438	1.806576	3.603227	2	2.043412	2.053914	2.027877	3.824140
3	1.944964	1.835198	1.809365	3.602802	3	2.041687	2.054201	2.035003	3.823483

ran on machine shirov

ELE709 - Real-Time Computer Control Systems
Lab 3 - POSIX Threads and Concurrent Programming (Week 2)

Name: Toni Pano #500822828 Section 3

Timing information should be obtained by running the required programs for the Exercises on a workstation in ENG413

1. **Exercise 3.4**

   (a) Repeat Exercises 3.2 and 3.3 with the `load` program running concurrently. Record the results in Table A.2 below.

   (b) Are the timing results similar to those obtained when the `load` program *wasn't* running concurrently? Explain why (or why not).

   Most timing results are slower with the load program running concurrently, however the relationship between exercise 3.2 and 3.3 stays similar with and without the load. The timing results are slower because the load adds 3 more threads that must run concurrently with each thread in Exercise 3.2 and 3.3. This means each thread will have to be interrupted periodically at a much higher frequency than with no load running. The overhead from switching between threads more frequently may add a more significant timing delay to the elapsed time seen in all 4 threads with a load.

2. **Exercise 3.5** Demonstrate your concurrent matrix multiplication program to the TA.

✓

Table A.2: Execution Time per Iteration (With Load) in nanoseconds

Exercise 3.2 (With Load)					Exercise 3.3 (With Load)				
	+	−	×	÷		+	−	×	÷
1	1.919302	1.922006	1.918085	3.828969	1	2.088095	2.196231	2.189546	4.058175
2	1.918479	1.920696	1.918024	3.828215	2	1.916402	2.124787	2.128317	5.341322
3	1.917910	1.921004	1.917612	3.828464	3	1.915774	2.181207	2.177643	5.327664

ran on machine sokolov