**Operating Systems: Problem Set 1**

**Part 1: System Calls: The Kernel Overhead**

3,000,000 iterations of the system call take 6 seconds. This means that one system call takes 2 microseconds per iteration. 40,000,000 iterations of the procedure call take 3 seconds. This accounts to 0.075 microseconds per iteration.

The system call takes about **26.67** times as long as a procedure call.

**PART 2: Threads**

a) code:

Code is shared between the threads. The code remains constant and therefore it would be wasteful to keep redundant copies for every thread. The process spawns new threads as it goes along its code

b) data:

Data is shared between the threads. Constants and globals are shared becasue they do not change and therefore it would be wasteful to keep redundant copies for every thread.

c) files:

Files are shared between the threads. There is only one file descriptor table per process, and it's shared among all the threads. We want to be able to access one file with multiple threads.

d) registers:

Registers are not shared between the threads. Each thread of execution has its own separate thread context and needs to be able to rely on its registers, which hold its current working variables. If threads are shared, overwriting of registers would not be prevented.

e) program counter

Program counter are not shared between the threads. Each thread has its own program counter that keeps track of which instruction to execute next. If program counters are shared, the threads would be unable to know where in their code they were.

f) stack

Each thread has its own stack, which contains the execution history, with one frame for each procedure called but not yet returned from. Each thread will generally call different procedures and a thus have a different execution history. This is why is a thread needs its own stack as otherwise, it would possibly get some other thread’s return values. If shared, it would not be possible to access methods in threads and would need pushing/popping.

g) PCB - process control block

The PCB is shared as each process has a single PCB. The PCB contains the kernel information about the process, which is shared among threads. This issame for all threads, does not need to be duplicated. If PCB is not shared, it would mean different processes.

**PART 3: Concurrency and Threads**

Code in **Cachetest.c** file