# CSE 451 Problem Set #1: Context Switching
# Due Date: Thursday, October 14, in section

0. This assignment can be done either in the Nachos teams of 2-4, or if you haven't formed a team yet, it is ok to do it individually. If you do it as a team, make sure each of you really understands the solution.

1. The point of this assignment is to make sure you **really** understand the concept of context switching. Write down the sequence of procedure calls and returns, along with changes to the ready list, that would occur in Nachos if the "main" thread in Nachos were to run the Thread::SelfTest2 code listed below. For each procedure call/return, give the name of the thread doing the call/return, the procedure name, and the values of any significant arguments. You only have to list calls and returns to routines **defined** in thread.cc, scheduler.cc, and switch.s, along with calls to Interrupt::SetLevel() -- in other words, you can ignore ASSERTs, DEBUG statements, calls to list routines, calls made internal to the interrupt simulation, any trivial inline functions, etc. You should also assume that timer interrupts are disabled -- there is no preemption; you should also assume that there are no threads on the ready list when Thread::SelfTest2 is called.

For example, your solution should begin:

readylist = NULL; running "main"
In "main", calling Thread::SelfTest2()
In "main" calling Thread::Thread("child 1")
In "main" returning from Thread::Thread("child 1")

It is acceptable (even encouraged) for you to generate your solution automatically. The code to be run is:

```
void
ChildFunction() {
     kernel->currentThread->Yield();
}

void
Thread::SelfTest2() {
    Thread *t1 = new Thread("child 1");
    Thread *t2 = new Thread("child 2");

    t1->Fork(&ChildFunction, t1);
    t2->Fork(&ChildFunction, t2);
    kernel->currentThread->Yield();
}
```