

## Lab #2: pthreads Introduction

### Overview

The purpose of this lab is to introduce pthreads and to explore simple synchronization and problem decomposition.

### Part 1: pthread\_create

Write a program named `p_test`. This program must take a single integer, `N`, as a command-line argument (look at `atoi` to convert a string into an integer). This first thread will create a second thread (the main thread is the first). The second thread must print the odd numbers from 1 to `N` (inclusive) while the first thread prints the even numbers from 1 to `N` (inclusive). After printing, the first thread should wait for the second thread to terminate.

For the odd numbers, use “`%d\n`” as the format string for `printf`. For the even numbers, use “`\t%d\n`” as the format string for `printf`.

### Part 2: Synchronization

Starting with the program from part 1, write a program named `m_test`. This program will behave as before, but must use synchronization to guarantee that the threads will take turns printing. First, the main thread should print 0. Then the second thread should print 1. Printing continues with each thread printing only a single number each turn.

There are a few ways to organize your solution, but it must use condition variables.

### Demonstration

Demonstrate your solutions to the instructor in lab.