CS 3430/5430 Homework 1 Due: Friday, Jan. 27 (3 PM)

From the on-line textbook, pp. 39-41 do:

- Ch 2 Exercises (pencil-and-paper): 2.3 and 2.4
- Programming Project: 2.1 (C/C++) **or** 2.2 (Java) **or** other programming language

Exercises: 2.3

Suppose thread A goes through a loop 100 times, each time performing one disk I/O operation, taking 10 milliseconds, and then some computation, taking 1 millisecond. While each 10-millisecond disk operation is in progress, thread A cannot make any use of the processor. Thread B runs for 1 second, purely in the processor, with no I/O. One millisecond of processor time is spent each time the processor switches threads; other than this switching cost, there is no problem with the processor working on thread B during one of thread A's I/O operations. (The processor and disk drive do not contend for memory access bandwidth, for example.)

(a) Suppose the processor and disk work purely on thread A until its completion, and then the processor switches to thread B and runs all of that thread. What will the total elapsed time be?

(b) Suppose the processor starts out working on thread A, but every time thread A performs a disk operation, the processor switches to B during the operation and then back to A upon the disk operation's completion. What will the total elapsed time be?

Exercises: 2.4

Consider a uniprocessor system where each arrival of input from an external source triggers the creation and execution of a new thread, which at its completion produces some output. We are interested in the response time from triggering input to resulting output.

(a) Input arrives at time 0 and again after 1 second, 2 seconds, and so forth. Each arrival triggers a thread that takes 600 milliseconds to run. Before the thread can run, it must be created and dispatched, which takes 10 milliseconds. What is the average response time for these inputs?

(b) Now a second source of input is added, with input arriving at times 0.1 seconds, 1.1 seconds, 2.1 seconds, and so forth. These inputs trigger threads that only take 100 milliseconds to run, but they still need 10 milliseconds to create and dispatch. When an input arrives, the resulting new thread is not created or dispatched until the processor is idle. What is the average response time for this second class of inputs? What is the combined average response time for the two classes?

(c) Suppose we change the way the second class of input is handled. When the input arrives, the new thread is immediately created and dispatched, even if that preempts an already running thread. When the new thread completes, the preempted thread resumes execution after a 1 millisecond thread switching delay. What is the average response time for each class of inputs? What is the combined average for the two together?

Programming Project 2.1

If you program in C, read the documentation for pthread_cancel. Using this information and the model provided in Figure 2.4 on page 26, write a program where the initial (main) thread creates a second thread. The main thread should read input from the keyboard, waiting until the user presses the Enter key. At that point, it should kill off the second thread and print out a message reporting that it has done so. Meanwhile, the second thread should be in an innite loop, each time around sleeping ve seconds and then printing out a message. Try running your program. Can the sleeping thread print its periodic messages while the main thread is waiting for keyboard input? Can the main thread read input, kill the sleeping thread, and print a message while the sleeping thread is in the early part of one of its ve-second sleeps?

Programming Project 2.2

If you program in Java, read the documentation for the stop method in the Thread class. (Ignore the information about it being deprecated. That will make sense only after you read Chapter 4 of this book.) Write the program described in Programming Project 2.1, except do so in Java. You can use the program shown in Figure 2.3 on page 25 as a model.

Programming project 2.1 or 2.2 should be submitted electronically as follows:

In a browser go to: **https://math-cs.cns.uni.edu/~schafer/submit/which_course.cgi**

- You will be asked to enter your AD-ITS Username and Password
- You will be asked to select your "Course and section number" which is "CS 3430/5430, Operating Systems, Fienup" and then click the "Continue" button
- You will be asked to select the homework you are submitting and then click the "Continue" button
- You may be asked how many extra files you would like to submit in **addition** to hw1.zip.
- You can now browser for the file(s) you want to submit before clicking "Continue".
- The system tells you which files got uploaded (i.e., submitted). If you discover a mistake later (but before the deadline), you can repeat this process and replace the previously submitted file.

If you ever have problems using the submission system (e.g., the deadline time has past), you can always email the zipped file to me at: fienup@cs.uni.edu.