CS211 **Labs 5: Processes**

This lab is a set of exercises, all based on ones in Chapter 5 of the textbook; see http://pages.cs.wisc.edu/~remzi/OSTEP/.
Since the use the `fork()` system call, and other UNIX-related system calls, you'll need to complete these exercises using in suitable online compiler; `code::blocks`, with the *mingw* compiler, is not sufficient.

1. Choose a suitable online compiler. Verify that you can run sample programmes from Week 7, in particular, `02WhoAmI.c` and `08Pipes.c`.

2. Write a program that calls `fork()`. Before calling `fork()`, have the main process declare an initialise an `int` variable $x = 100$. What value is the variable in the child process? What happens to the variable when both the child and parent change the value of x?

3. If a process opens a file, does a child process have access to it? What happens if they both try to write to the file at the same time? To answer this, download `02fopen.c` from http://www.maths.nuigalway.ie/~niall/CS211/lab5. Notice the use of the `fflush()` system call; how does the output change if that is removed?

4. Write another program using fork(). The child process should print "hello"; the parent process should print "goodbye". You should try to ensure that the child process always prints first; can you do this without calling `wait()` in the parent?

5. Before answering the next question, read, compile and run `04WaitAndCount.c`. See http://www.maths.nuigalway.ie/~niall/CS211/Week07/

6. (From OSTEP) Write a program that uses `wait()` to wait for the child process to finish in the parent. What does `wait()` return? What happens if you use `wait()` in the child?

7. Find out what the `waitpid()` function does. Write a programme that exhibits that.

You do not have to submit your work from this lab. But we will develop these ideas into an assignment next week.