**UNX511/DSP912 – Lab 5: Monitoring Process Memory Usage via fork()**

**Due: Monday, February 18th, 11:59PM**

Information on all processes running on a Linux system is contained within the **/proc** directory. In Lab 1 you wrote a C/CPP program that scanned all processes running on the system and reported on those which had used more than 10Mbytes of resident memory. You did this by going through each process directory in **/proc** and looking at the file **status**.

Inside the file **status** are variables such as **Name:** (the process name), **Pid:** (the process pid), **VmSize:** (Virtual Memory) and **VmRSS:** (Resident memory).

In Lab 1 your C/CPP program scanned every directory in **/proc** that represented a process (each directory that begins with a number represents a process). You opened the file **status**. If **VmRSS:** was greater than 10000 kB, you printed out the process name, the process id and the memory used.

In this lab you will do the same as lab 1 with one exception – you will break this task up into four parts using **fork()**. Once you get the number of processes in the system, you will:

1. Break the number of processes into 4 chunks.
2. **fork()** four times so each child handles one of the four chunks.
3. The parent should return after having created 4 children in order to let the children do all the work.

For example, if there are 100 processes running in the system, the first child should handle the first 25 processes, the second child should handle the next 25 processes, the third child the next 25 processes, and the fourth child the last 25 processes. These children should operate in parallel with each other. The parent should not handle any of the processes.

1. Before a child starts handling its chunk of processes, the child should output the child number and the pid of the child, For instance:

$ **child #1 pid=1234**

$ **child #2 pid=1235**

$ **child #3 pid=1236**

$ **child #4 pid=1237**

1. What order do these children execute in and why (put your answer in your Lab5.cpp)?

The source code for Lab 1 has been provided for you with comments. It has been renamed as Lab5.cpp and can be retrieve [here](https://scs.senecac.on.ca/~miguel.watler/courses/unx511/Lab5/Lab5.cpp). Please modify Lab5.cpp to fulfill the requirements of the lab.

**Lab Submission:**

Email me your Makefile, and your Lab5.cpp to:

miguel.watler@senecacollege.ca

**NB: My last name is Watler, not Walter.**