CS2106 Introduction to Operating Systems
Lab 1 – UNIX System Calls

Each exercise below should be submitted as a single C file with the name given. Zip up all files into a single ZIP file with the name Axxxxxx.zip, where Axxxxx is your matric number. You will then email the files to your respective lab TA. Marks for each question are indicated in square brackets.

In general you should write these programs on the Sunfire system. You may however use your own UNIX system like LINUX or OSX. You will not be able to write these programs on a Windows system.

REMEMBER TO INCLUDE YOUR NAME AND MATRIC NUMBER IN EACH PROGRAM!

SUBMISSION DEADLINE: 21 February 2014 5 pm.

Question 1 (2 marks)

Write the following program in a file called lab1q1.c:

Use the getpid, getppid, and getcwd system calls to write a C program that prints its own process ID, the process ID of its parent, and its current working directory. Full documentation of these system calls can be found at:

http://www.gnu.org/software/libc/manual/html_node/index.html

Question 2 (2 marks)

Write the following program in a file called lab1q2.c

Let us now consider a different program with the same output as the exercise above. The syscall function available in the libc library (documentation at the same link as above) performs a direct system call, as opposed to providing a wrapper. In general, the use of this function is not recommended, as the syntax of the parameters varies from system to system. However, it is relatively easy to use when it has only one parameter. Thus, we can replace the three calls in the exercise above by: syscall(SYS_getpid), syscall(SYS_getppid), and syscall(SYS_getcwd). Perform these replacements in the program that you wrote for the previous exercise, and compile and run your it to check that the output is the same.

Question 3 (2 marks)

Write the following in a file called lab1q3.c

For most systems nowadays, a call to malloc(1024*1024*1024) will succeed, effectively allocating 1GB of memory for your program. Write a program that finds out how many Gigs of memory can be allocated to a program on your system (do not bother with reporting fractions of a Gig).