Operating Systems Lab Assignment

Lab 1 and 2: Introduction to OS and Process Basics

A) Introduction to OS:

Objective: The aim of this lab assignment is to refresh your knowledge in basic shell commands, and shell scripting. A few commands are mentioned below. Try these commands (and any other you wish to), and for each command, report (note down) the following:

a) correctly typed command,
b) results, and
c) observations (if any)

Note:

1) $ indicates the prompt on opening terminal / shell in your Linux machine
2) Use manual entry pages in Linux (man) for help on using commands – syntax and options available
3) OS: Linux / Linux flavors – Debian / Ubuntu / Fedora, etc.

Commands:

1) $date
2) $cal
3) $echo <text>
4) $man <command name>
5) $ls
6) $pwd
7) $mkdir
8) $cd
9) $rmdir
10) $cat
11) $sort
12) $cp
13) $mv
14) $rm
15) $wc
16) $head
17) $tail
18) $more
19) $sort
20) pipe (|)
21) $tr
22) $chmod [Both symbolic and absolute representations]
23) $chown
24) $chgrp
25) Write a shell script to generate the Fibonacci series

B) Process Basics

Objective: The aim of this lab assignment is to get you comfortable with the basics of processes - finding out which processes are running, their metadata – various information displayed in the process

listing, memory allocated, CPU times, process states, creating and deleting processes and initial forays into process and resource management

Tasks / Programs:

1) Use shell commands to list running processes, to view the processes that are consuming the most resources, search specific processes, change the priority (nice value) of a process, and kill a process. Here are some commands you can try:

a) $ps
b) $pstree
c) $top
d) $pgrep
e) $renice
f) $kill
g) $xkill

* Note down the output and the observations for all the commands.
* For commands that display process information, note down which processes are consuming the most amount of memory, which consume the least, and what are the states of the top processes

2) Write a simple C program that creates a process using the fork() call. Use the getpid() and getppid() calls to print the child and parent id in created child process, and return value of fork call in parent process

3) Write a simple C program to create a process using the exec() call. What is the difference between a fork() and an exec() call?

4) Write a program to find out the number of processes and their states, like CPU time, submission time, memory size of the process etc. at any instant. Store the information in a table structure for further processing

5) Given a set of processes (P1, P2, ..., Pn) and resources (R1, R2, ..., Rn), at any instant, the ser of requests (Pi to Rj) and the set of assignments (Ri to Pj) are given:

a) Find out that no single resource is assigned to multiple processes
b) Find out that the processes is not making repeated requests for the same resource