Project8:

Part1: Pgrep, pkill, kill and jobs

**In these exercises, we will be working with jobs running on the system. To follow this exercise, please open two SSH terminals into the same system. Log in as the***root***user in one and a non-root user in the other. The non-root username in this exercise will be referred to as***user***student**

**Use Rocky8 system or redhat 8**

Exercise 1

1. As the *root* user, create a job running in the background of your current terminal. Execute the following script for that program process to be created:

(while true; do echo "My program" > ~/output.file; done) &

2. View the current jobs running in the background of your terminal.

3. Stop the process from running, *without* killing the process, using the kill command.

4. View the stopped jobs in the background.

5. Start the process again using the kill command.

6. Kill the process without allowing any blocking of the kill command.

Exercise 2

1. Download and install the httpd service.

2. Start the httpd service (or ensure that it is running).

3. As the *root* user, grep for all processes that are running as the *root* user and display the process names.

4. As the *user* student, start the vi program at the terminal.

5. As the *root* user, in your second terminal window, grep for all processes running under the user "student" and include the process names.

6. As the *root* user, grep for the "httpd" process.

7. As the *root* user, kill all of the "student" user's processes and boot that user out the system.

Part2: Nice, renice and ps

Use redhat8 or rocky 8

1. Ensure that you have the httpd package installed on the system.
2. Ensure the httpd service is *not* running.
3. Start the httpd service with the most favorable nice possible.
4. View the current nice of the httpd service using the ps command and grep command together.
5. Renice all httpd processes and set the nice level to 0.

Part3: Monitoring and Calculating CPU Load Averages

Use redhat8 or rocky 8

1. View the system uptime and load average.
2. View the system uptime and load average in such a way that it also shows what users are logged in to the system and what the user is doing.
3. Using the proc file system and wc, display the number of processors your system has. This is important to calculate the load average of the system.
4. Calculate the 1, 5, and 15-minute CPU load averages for the system.

Part4: Working with top

Run below command on your terminal

[root@localhost]# (while true; do echo -n "my program" >> /dev/null; done;) &

1. On a Rocky 8 lab server start the top program.
2. The top program shows all running processes on the system but sorts them. Using your keyboard, browse up and down to view the processes.
3. Sort all processes by memory percentage.
4. Sort all processes by CPU usage.
5. Renice the process for the script started at the beginning of the exercise. The command is  "bash", as displayed in the top program. Set the nice level to -20.
6. Kill the bash script using top.