CSI 3336 Systems Programming
Project 3

Shell Scripting

Shell scripts are used to help control many parts of a running a Unix system. We can also use them to simplify some routine user tasks. That's what you are going to do for this assignment.

We will be using the command line parameters for each of the scripts we write. If the proper number of command line parameters are not given or invalid data is given, we must present the user with an error message. Generally speaking, error messages in Linux exhibit a particular format and are written to standard error not standard out. The general format of an error message displays the command and any parameters. Parameters located within brackets generally indicate that the parameters are optional. Here is a simplified example of chmod's error message.

usage: chmod [-fhv] file

activity.sh

You're going to write a simple shell script called activity.sh. This script will go through all the files starting in the folder provided on the command line and all subdirectories and total up the sizes of files of various ages. You will report the number of files and the total size for files in three categories. We'll say active files are those that have been modified in the last 24 hours. Recent files are those that have been modified in the past 3 days, but not within the last 24 hours. Idle files are those that have not been modified in the last three days. When ran, your script will print out a report like:

```
/home/csi/m/maars
active: 11 (4281 bytes)
recent: 20 (92310 bytes)
idle: 135 (129562 bytes)
```

This script will need to use the find command to find files of various ages. You can use loops to look at each file individually and variables to accumulate the various totals. As always, be sure your program is well commented and consistently indented.

genRandom.sh

Shell scripts can also be used to provide some sort of service. Everyone of us have found the need to create test files for our programs. Much of the time, these files just need some data, which makes random values more than acceptable. Sure, we could

obtain this functionality with a program. If we find it super useful, we should write it as a program. But, we are going to write an additional script.

genRandom.sh should generate and display to standard out a space delimited list of random numbers given zero, one, or three command line parameters. If no parameters are given, genRandom.sh will provide a list of 10 random values with a range of 0 - 9999 that is inclusive. The first optional parameter of genRandom.sh indicates the number of random values that it should generate. The second optional parameter indicates the lower bound while the third optional parameter indicates the upper bound. All random values should fall within the lower and upper bounds inclusively.

For this script you are required to obtain your random values from the environment variable named RANDOM. You can obtain a random sequence of values by repeatedly accessing RANDOM.

Put a copy of your shell scripts in your new bin directory so you can run it whenever you want. When you are done, submit an electronic copy of both of your scripts (via a submission script) and turn in a printout on the day it is due.