OS1 Lab 5.2 - Linux Review Exercises

Sample Solutions

Dr. Martin O'Connor

26 February, 2025

File Permission Exercises

In this section, you will complete exercises to create a file and modify the file's permissions.

1. Create a new file called **text.txt** using the `touch` command .
2. What are the permissions of the file **text.txt**?

   Answer:
   `-rw-r--r--`

Please complete the following exercises to change the permissions of the file **text.txt** using the command sequence `chmod mode filename` where mode is created by concatenating characters from *who*, *opcode* and *permission*. Please write the commands you used as your **Answer**:

1. Allow the owner read access to the file and deny everyone else access to the file.

   Answer:

   ```
   $ chmod u+r,go-rw test.txt
   OR
   $ chmod u=r,go= test.txt
   ```

2. Allows the owner and world to read the file but deny all access to the group.

   Answer:

   ```
   $ chmod o+r test.txt
   OR
   $ chmod uo=r,g= test.txt
   ```

3. Deny access to the file for the owner, group and world.

   Answer:

```
$ chmod uo-r test.txt
OR
$ chmod ugo= test.txt
```

4. Grant the owner read and write access and grant the group and world read only access.

   Answer:

```
$ chmod u+rw test.txt
OR
$ chmod u=rw,go= test.txt
```

Please complete the following exercises to change the permissions of the file **text.txt** using the command sequence `chmod octal-mode filename` where *octal-mode* is created by concatenating three digits from 0 to 7 inclusive. Please write the commands you used as your **Answer**:

1. Allow the owner read access to the file and deny everyone else access to the file.

   Answer:

```
$ chmod 400 test.txt
```

2. Allows the owner and world to read the file but deny all access to the group.

   Answer:

```
$ chmod 404 test.txt
```

3. Deny access to the file for the owner, group and world.

   Answer:

```
$ chmod 000 test.txt
```

4. Grant the owner read and write access and grant the group and world read only access.

   Answer:

```
$ chmod 644 test.txt
```

File and Text Exercises

Create a file called **capital.txt** with the following five lines:

```
Dublin is the capital city of Ireland.
Ireland is viewed as the best country in the world by many people.
```

```
Dublin has a population greater than 1 million people.
There are many places to see and visit in Dublin.
The largest university in Ireland is TU Dublin.
```

Create a second file called **cities.txt** with the following contents:

```
Dublin
Cork
Belfast
Galway
Waterford
Kilkenny
```

Please complete the following exercises and write the commands you used as your answer.

1. Display the cities in the **cites.txt** file in reverse sorted order.

   Answer:

   ```
   $ sort -r cities.txt
   ```

2. Create a new file called **sortedcites.txt** that contains the cities in **cites.txt** in sorted order. Hint: *Use the sort command and redirection*.

   Answer:

   ```
   $ sort cities.txt > sortedcities.txt
   OR
   $ sort < cities.txt > sortedcities.txt
   ```

3. Display just the last two lines in the file **capital.txt**. Hint: *Use the tail command*.

   Answer:

   ```
   $ tail -2 capital.txt
   OR
   $ cat capital.txt | tail -2
   ```

4. Display just the first three lines in the file **capital.txt**. Hint: *Use the head command*.

   Answer:

   ```
   $ head -3 capital.txt
   OR
   $ cat capital.txt | head -3
   ```

5. Display just the lines in the file **capital.txt** that contain the word `Dublin`. Hint: *Use the grep command*.

   Answer:

```
$ grep Dublin capital.txt
OR
$ cat capital.txt | grep Dublin
```

6. Display just the lines in the file **capital.txt** that contain the word `dublin` case-insensitive.

   Answer:

```
$ grep -i dublin capital.txt
OR
$ cat capital.txt | grep -i dublin
```

7. Display just the lines in the file **capital.txt** that **do not** contain the word `dublin` case-insensitive.

   Answer:

```
$ grep -iv dublin capital.txt
OR
$ cat capital.txt | grep -iv dublin
```

8. Display just the count of the lines in the file **capital.txt** that **do not** contain the word `dublin` case-insensitive.

   Answer:

```
$ grep -ivc dublin capital.txt
OR
$ grep -iv dublin capital.txt | wc -l
```

9. Display just the line number(s) in the file **cities.txt** that contains the word `cork` case-insensitive.

   Answer:

```
Correct answer in this case given only one digit in the line numbers:
$ grep -in cork cities.txt | cut -c02

Correct answer in all cases regardless of the number of lines in the
file:
$ grep -in cork cities.txt | cut -d':' -f-1

-d specifies the delimiter separating the input token fields.
-f <from>-<to> specifies start and end fields to process
```

10. Display just the first character of each line in the file **cities.txt**. Hint: *Use the cat and cut commands with piping*.

    Answer:

```
$ cat capital.txt | cut -c01
```

11. Display just the second last line of the file **capital.txt**. Hint: *Use the cat, tail and head commands with piping*.

    Answer:

```
$ tail -2 capital.txt | head -1
OR
$ cat capital.txt | tail -2 | head -1
```

12. Display a listing in long format of all files in the current working folder whose filenames begin with the letter k and end with the extension .txt

    Answer:

```
$ ls -l k*.txt
```

13. Write the command that would run the program `myprogramme`, discarding all output generated to the null device.

    Answer:

```
All output means both stdout and stderr output.  Thus, answer is:
$ ./myprogramme 2> /dev/null > /dev/null
```

14. Display the output of the command `ps -aux` *both* to the screen *and* redirect the stdout to the file **processes.txt**.

    Answer:

```
$ ps -aux | tee processes.txt
```

15. Append the string "test" to the file **/etc/passwd** redirecting all error messages to the file **error.txt** in the current folder. Hint: *Use the echo command with redirection*.

    Answer:

```
$ echo "test" 2> error.txt > /etc/passwd
```

16. Delete the file **/etc/passwd** (without prompting the user for confirmation) redirecting all error messages generated by appending them to the file **error.txt** in the current folder.

    Answer:

```
$ rm -f /etc/passwd 2>> error.txt
```

17. The command `seq` generates a sequence of numbers. For example, the command `seq 5` generates a sequence of numbers from 1 to 5. Try it out for yourself.

    1. Create a new file called **numbers.txt** with the sequence of numbers from 1 to 4 each on a separate line. Hint: *Use the seq command with redirection*.

       Answer:

```
$ seq 4 > numbers.txt
```

2. Now, using one command, appended the sequence numbers from 1 to 6 to the end of the file **numbers.txt**., each number on a separate line.

   Answer:

```
$ seq 6 >> numbers.txt
```

3. Display the contents of the file **numbers.txt** sorted in ascending order.

   Answer:

```
$ sort -n numbers.txt
OR
$ cat numbers.txt | sort -n
```

4. Display the contents of the file **numbers.txt** sorted in ascending order, with duplicates removed. Hint: *Use the cat, sort and uniq commands with piping*.

   Answer:

```
$ sort -n numbers.txt | uniq -u
OR
$ cat numbers.txt | sort -n | uniq -u
```

5. Display how many times each line is duplicated in the file **numbers.txt**.

   Answer:

```
$ sort -n numbers.txt | uniq -c
OR
$ cat numbers.txt | sort -n | uniq -c
```

6. Display only the unique (non-duplicate lines) in the file. **numbers.txt**.

   Answer:

```
$ cat numbers.txt | sort -n | uniq -u
```

Exercises on Processes

1. Start five processes in the background as indicated below:

```
$ sleep 1000 &              P1
$ sleep 2000 &              P2
$ sleep 3000 &              P3
$ sleep 4000 &              P4
$ sleep 5000 &              P5
```

   Please complete the following exercises, and write the commands and/or keystrokes you used as your answer.

2. Terminate process P1 without using the kill command.

   Answer:
   Run the command
   `$ jobs`
   to obtain the job number of process P1.
   Bring the process to the foreground by running the command
   `$ fg %jobnumber`
   Kill the process by pressing `Ctrl-C`

3. Identify the process identifier (PID) of process P2.

   Answer:
   Run the command
   `$ ps -x`
   to list all processes initiated by the current user, including process P2.
   To more quickly identify the process identifier of process P2, run the command
   `$ ps -x | grep -i "sleep 2000"`

4. Terminate process P2 using the kill command.

   Answer:

```
$ kill 2334                    [Replace the number with the correct PID]
```

5. Suspend process P3.

   Answer:
   Run the command
   `$ jobs`
   to obtain the job number of process P3.
   Bring the process to the foreground by running the command
   `$ fg %jobnumber`
   Suspend the process by pressing `Ctrl-Z`

   OR

```
$ kill -s stop 2337          [Replace the number with the correct PID]
```

6. Bring process P4 to the foreground and then terminate it.

   Answer:
   Run the command
   `$ ps -x`
   to list all processes initiated by the current user, including process P4.
   To more quickly identify the process identifier of process P4, run the command
   `$ ps -x | grep -i "sleep 4000"`
   Then run the command
   `$ kill 2342                [Replace the number with the correct PID]`

7. Terminate process P5 by force using the kill command.

Answer:
Run the command
`$ ps -x`
to list all processes initiated by the current user, including process P5.
To more quickly identify the process identifier of process P5, run the command
`$ ps -x | grep -i "sleep 5000"`
Then run the command (note the command line option -9)
`$ kill -9 2348 [Replace the number with the correct PID]`

8. Terminate process P3 without using the kill command.

Answer:
Run the command
`$ jobs`
to obtain the job number of process P3.
Bring the job to the foreground by running the command
`$ fg %jobnumber`
Kill the job by pressing `Ctrl-C`