Shell Scripting

A. Some common Bash commands and text editors

Task 1

In this task, you will use the *pwd* command to view your working directory.
To display your current path:
1. Type **pwd** and press **Enter**.
2. What is your current directory path?

Task 2

The *ls* command is one of the most useful commands. In this task, you will start by using *ls* to view your working directory. Next, you use *ls* with an argument to view a file and then a directory. For a more complete listing of information about the contents of a directory, you use the *-l* option, and finally you use the *-a* option to include hidden files in a directory listing.
To see a list of files and directories in your current working directory:
1. Type **ls** and press **Enter**. You see a list of file and directory names.
To use the *ls* command with the *-l* option:
1. Type **ls -l**
2. Type **ls -l /** and press **Enter** to view the contents of the root file system directory.
To list hidden files in your home directory:
1. Type **clear** and press **Enter** to clear the screen.
2. Type **ls -a** after the command prompt and press **Enter**.

Task 3

To create new directories:
1. Type **cd** and press **Enter** to make certain you are in your home directory.
2. Type **mkdir new_dir** and press **Enter** to make a new directory called new_dir.
3. Type **ls** and press **Enter**. You see the new_dir directory in the listing.
4. Type **cd new_dir** and press **Enter** to change to the new directory. Now, you can create new files here

Task 4

Navigating a file system using the dot and dot dot options can save you typing time. In this task, you practice using both conventions. Make certain you are logged in to your own account for this task and not as root.
To use dot and dot dot to change your working directory:
1. If you are not in your home directory, type **cd** and press **Enter**.
2. Type **cd ..** and press **Enter**. The system takes you to the parent directory, which is /home.
3. Type **cd ..** and press **Enter**. The system takes you to the root file system directory

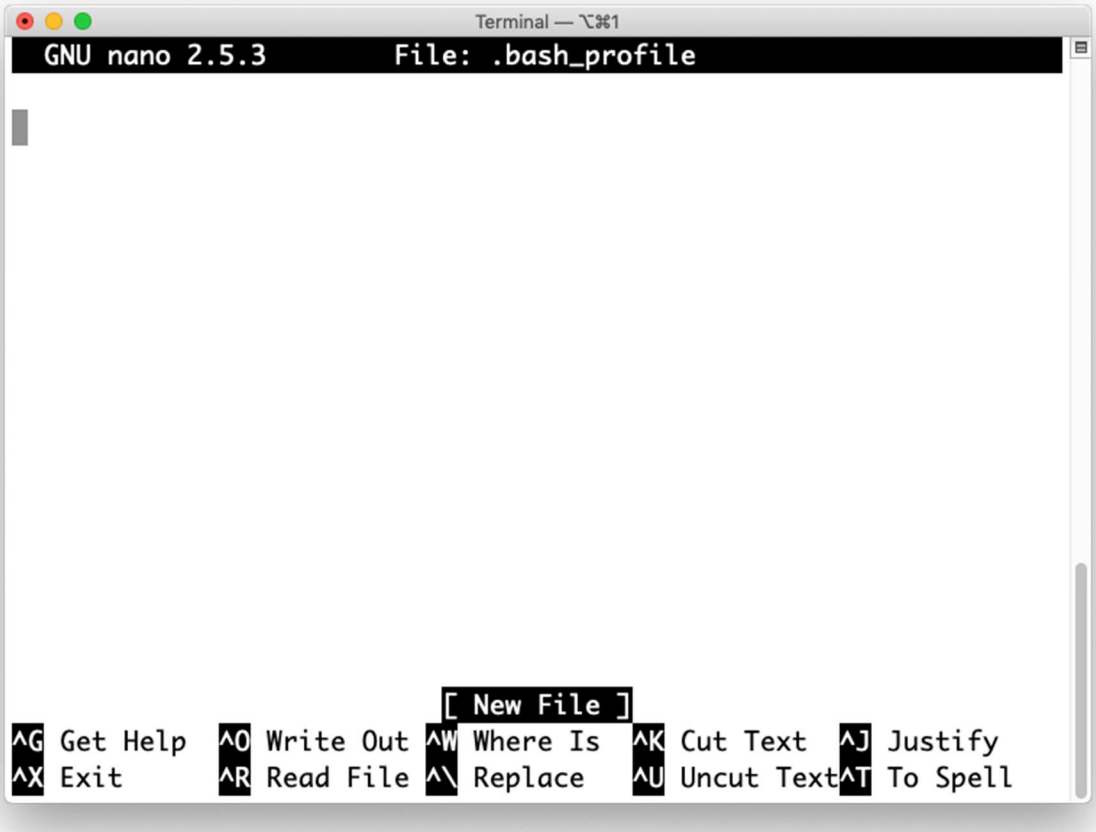
(/). Type **cd** and press **Enter**. The system takes you to your home directory.

Task 5

In this task, you start the nano editor by creating a file called textfiles to hold some basic comments about text files. This is simply a practice file to get you started learning the nano editor. It is generally best to learn on a file that is not important to your work. To open nano and create a new file, type *nano* followed by the new file's name.

To open nano and create a new file:

1. After the $ command prompt, type **nano textfiles** and press **Enter**. This starts nano and begins editing a new file called textfiles. Your screen should look similar to the one presented here.



In the upper-left corner of your screen, you see the cursor. The cursor indicates your current location in the file.

Check this link for info on how to use the nano editor.

https://www.cs.umb.edu/~ghoffman/linux/nano_text_editor.htm

Use this file to write what you learned so far about using the nano editor. Precede each line with # since you are writing comments about nano files.

Enter in a new line at the end of your file:

echo “I know how to use the nano editor.”

Save your file (Ctrl+O)
Exit the Editor (Ctrl + x)
Run the script by typing: ./textfiles
You should see printed in the screen:

I know how to use the nano editor.

Task 6

Create a new script called commands.sh
Type ls -l
Save your file (Ctrl+O)
Exit the Editor (Ctrl + x)
Run the script by typing: ./commands.sh
You should see the output of the ls -l command on you screen.

You should be able to see the two new files that you created: textfiles and commands.sh

Task 7

View the content of the file textfiles you just created in two ways:

1) In the shell by typing:
   cat textfile
2) By opening it in the editor again:
   nano textfile

Task 8

Remove the textfile you created

Exercises

1. Use the *ls* command to list the contents of your working directory on your system.
2. Use the *ls -l* command to view the contents of the root file system directory (/).
3. Create a directory called my_first. Change to my_first. Display its content with ls.
4. Are there any hidden files in your working directory?
5. Make a directory under your home directory called docs. Next, make a directory under the docs directory called spreads. What is the absolute path for the spreadsheets directory?

6. Change to your home directory. Use the *rmdir* command to delete the spreads directory. What happens?
7. Create a script named script1. Use the script to clear your screen and then display your current working directory.

B. Using variables

Task 1

This task enables you to use the defining and evaluating operators to learn how they work.

To create a variable, and assign it a value:
1. Type **DOG=Shepherd** and press **Enter**.
You've created the variable DOG and set its value to Shepherd.
To see the contents of a variable:
1. Type **echo DOG** and press **Enter**.
You see the word "DOG."
2. To see the contents of the DOG variable, you must precede the name of the variable with a $ operator. Type **echo $DOG** and press **Enter**. You see the word "Shepherd."

To use double quotation marks to set a variable to a string of characters containing spaces:
1. Type **MEMO="Meeting will be at noon today"** and press **Enter**.
2. Type **echo $MEMO** and press **Enter**.
You see the contents of the MEMO variable: Meeting will be at noon today.
To demonstrate how double quotation marks do not suppress the viewing of a variable's contents, but single quotation marks do suppress the viewing:
1. Type **echo '$HOME'** and press **Enter**.
You see $HOME on the screen.
2. Type **echo "$HOME"** and press **Enter**.

Task 2

In this task, you employ the *let* command to practice using arithmetic operators to set the contents of a shell variable. First, you use an expression with constants (no variables), and then you use an expression containing a variable.
To practice using the arithmetic operators:
1. Type **let X=10+2*7** and press **Enter**.
2. Type **echo $X** and press **Enter**. You see 24 on the screen.
3. Type **let Y=X+2*4** and press **Enter**.
4. Type **echo $Y** and press **Enter**. You see 32 on the screen.
5. Type **clear** and press **Enter** to clear the screen for the next task.

Task 3

In this task, you gain further experience in writing a very simple shell script using sequential logic. In these steps, you create the shell script, seqtotal.

To demonstrate sequential logic:

1. Type **nano seqtotal** and press **Enter**.
2. Type the following lines:

```
let a=1
let b=2
let c=3
let total=a+b+c
echo $total
```

3. Save the file and exit the nano editor
4. Execute by typing ./seqtotal

What is the output?

Task 4

In this task you will display the value of some built in variables. Type these command separately and press enter.

- Echo $HOME
- Returns user home directory

- Echo $PWD
- Return current directory

- Echo $MACHTYPE
- Returns machine type

- Echo $HOSTNAME
- Returns the systems name

- Echo BASH_VERSION
- Return version of BASH

- Echo $SECONDS
- Returns numbers of seconds the Bash has run

Exercises

1. Assign the variable *t* the value of 20. Next, assign the variable *s* the value of t+30. Finally, display the contents of *t* and s to verify you have correctly defined these variables.

T=20
S=$(($t+30) or s=$[$t+30]

Echo $s

Or
Let t=20
Let s=t+30
Echo $t $s

2. Create a script that declares three variables and prints their value
-one string variable
-one variable with spaces
-an integer
Use these variables inside other strings or variables and print them

3. Write a script that declares an integer d. Assign the value 4 to it.
Increase d by 4 and display the new value
Multiply d by 2 and display the new value
Let d=4
Let d=d+4
Let d=d*2
Echo $d

Or
D=4
D=$(($d+4))
Echo $d
D=$((d * 2))

Echo $d