Basic Commands in the Terminal: File & Folder Editing and Viewing

Author: Michelle Franc Ragsac (mragsac@eng.ucsd.edu)

Notebook Information:

This Jupyter Notebook contains the `UNIX` commands that were used in Module 1a: Bench to Terminal.
It's running with a `Bash` kernel so that all of the code blocks will have the output from running a command directly below them.

How can we create new files with `vim`?

Let's change into one of the new directories we created in the previous notebook and create a blank text file. To do so, we will use `vim` , one of many terminal-based text editor applications. Other terminal-based text editors include `emacs` and `nano` (you can see `nano` in action in this other [UNIX tutorial (http://korflab.ucdavis.edu/bootcamp.html)](http://korflab.ucdavis.edu/bootcamp.html) offered by UC Davis). Each of these applications has their own unique feel to them, but for this module, we'll be sticking with `vim` .

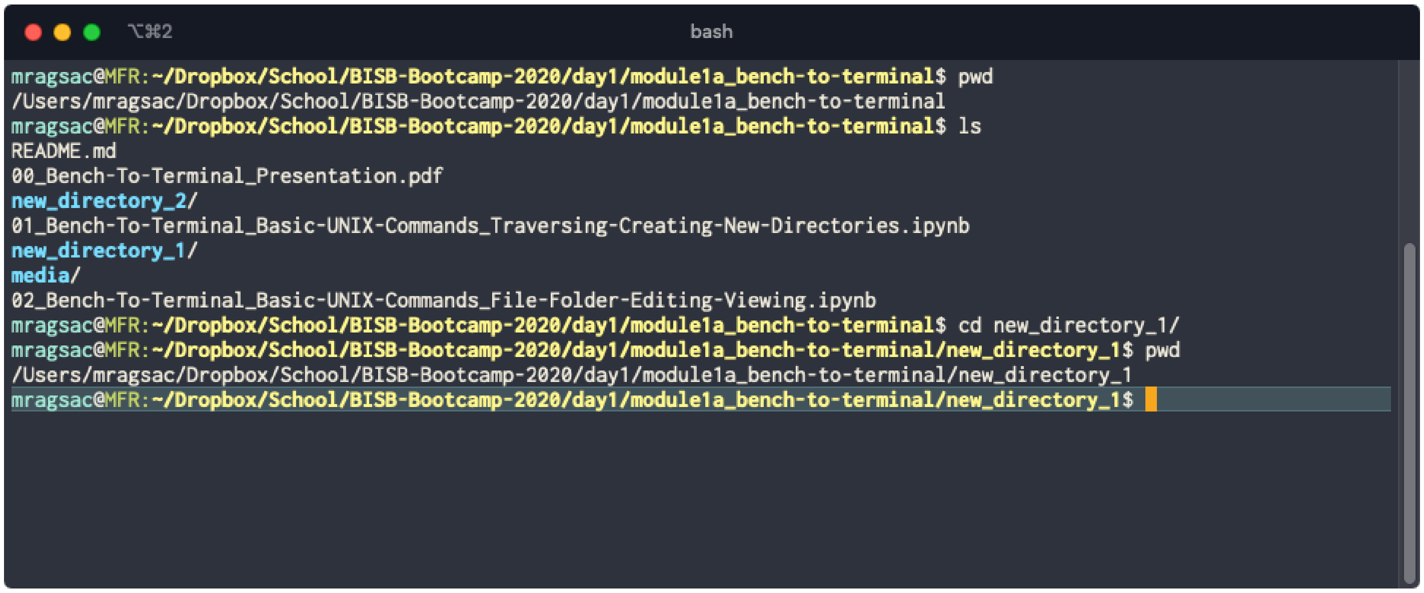
Note: I'll be running the code for this section ("Creating New Files with `vim` ") and the next section ("Editing New Files with `vim` ") on my own terminal application instead of within this Jupyter Notebook.
These two sections will contain screenshots of what you should see when following the commands in the terminal.

Changing into the `/home/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_directory_1/` Directory

First, let's start off by changing into one of the new directories we've generated from the last Jupyter Notebook, `new_directory_1` .

```
######################################
# Commands used within this section #
######################################

pwd               # prints the current working directory
cd                # changes the directory to the one listed
```



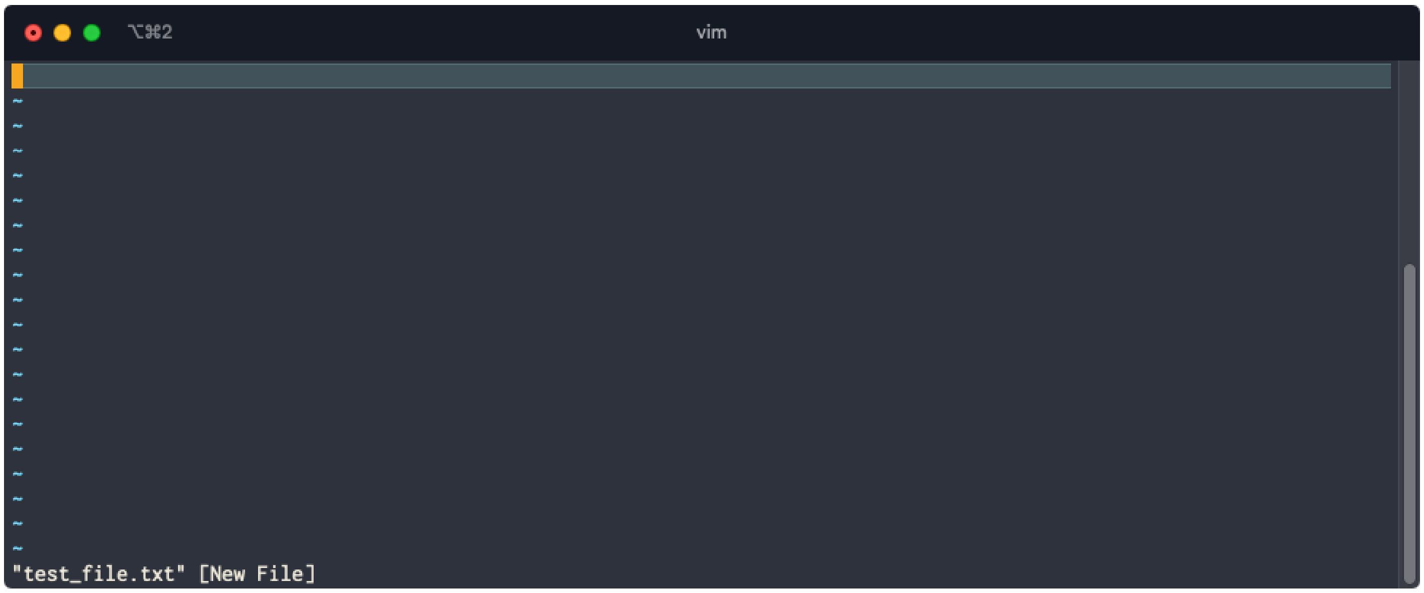
```
mragsac@MFR:~/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal$ pwd
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal
mragsac@MFR:~/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal$ ls
README.md
00_Bench-To-Terminal_Presentation.pdf
new_directory_2/
01_Bench-To-Terminal_Basic-UNIX-Commands_Traversing-Creating-New-Directories.ipynb
new_directory_1/
media/
02_Bench-To-Terminal_Basic-UNIX-Commands_File-Folder-Editing-Viewing.ipynb
mragsac@MFR:~/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal$ cd new_directory_1/
mragsac@MFR:~/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_directory_1$ pwd
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_directory_1
mragsac@MFR:~/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_directory_1$
```

Result of Running the Command `vim test_file.txt` in this Directory

Next, we'll create a new file called `test_file.txt` in the `vim` terminal-based text editor.

```
######################################
# Commands used within this section #
######################################

vim FILENAME    # creates a new file called FILENAME (variable) in the vim text editor
```



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"test_file.txt" [New File]
```

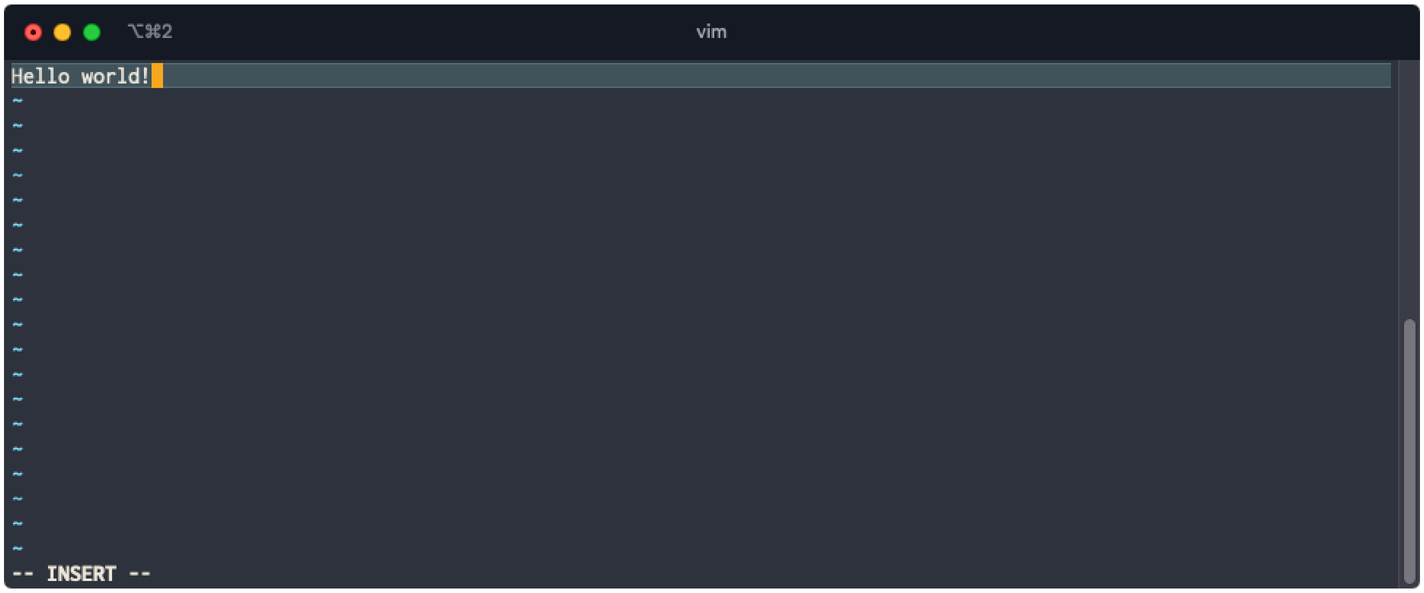
This blank screen contains the text file that you just created, `test_file.txt` . The file name is displayed in the bottom left-hand corner with a `[New file]` label.

Note: You can use the arrow keys to move the cursor around the file!

How can we edit new files created with `vim`?

You can edit the contents of the new `test_file.txt` file you're viewing in `vim` by pressing the `i` key on your keyboard. This will change the mode for `vim` to "INSERT mode". Afterwards, you can type in whatever you would like.

Editing the Contents of `test_file.txt` in `vim`'s `INSERT` Mode

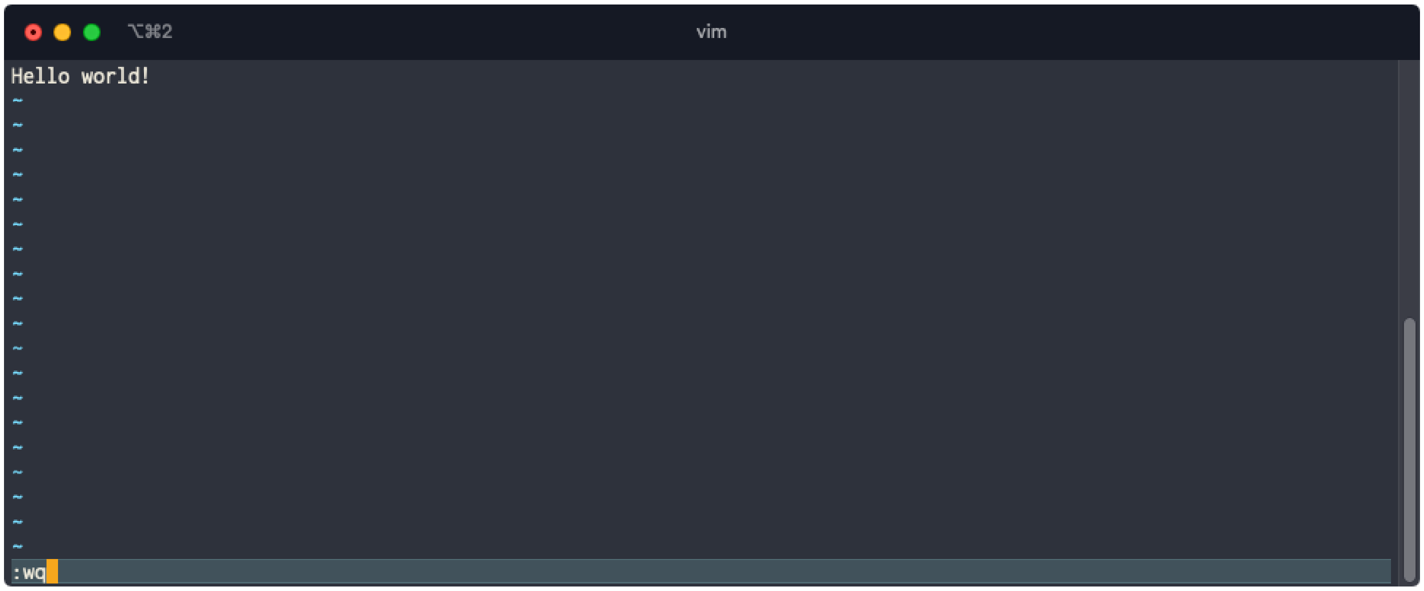


```
Hello world!
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

Saving the Contents of `test_file.txt` in `vim` then Exiting the Program with the `:wq` Command

To leave "INSERT mode" and prevent further editing of your file, you can press the `ESC` key.

Finally, to save and quit from the file, you can type in `:wq` (where `:` indicates a command for `vim` to follow, and `w` indicates you want to "write" or save the file, and then `q` indicates you want to "quit" right afterwards) then press the `ENTER` key.



```
Hello world!
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

After pressing the `ENTER` key, you should return back to the terminal.

How can we preview the contents of a file?

Say we want to preview the contents of the `test_file.txt`. We can preview the beginning, or "head", of the file with the command `head`:

In [1]:
```
# Change into the module directory, print the current directory (should be the same), th
en change into the new directory with the file we created
cd /Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal
pwd
cd new_directory_1
pwd

# List all of the files available in the folder
echo ""
echo "Listing all of the files available in the folder:"
ls

# Preview the beginning of the file with the head command
echo ""
echo "Previewing the contents of the file test_file.txt:"
head test_file.txt
```

```
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_di
rectory_1

Listing all of the files available in the folder:
test_file.txt

Previewing the contents of the file test_file.txt:
Hello world!
```

Note: Here, I've used the command `echo` to introduce some print statements to our `UNIX` commands and make the output a little bit more clear!

There are actually a couple of ways to view a file on the terminal! So far, we've used `head` to preview the "head" or beginning of the file. Here are some other commands we can use:

```
###############################
# Additional Command Examples #
###############################

cat FILENAME     # displays the entire contents of the file onto your screen
less FILENAME    # displays the entire contents of the file onto your screen in a way that c
an be scrolled through with the ENTRE key
tail FILENAME    # looks at the last few lines of a file (versus head)
```

I encourage you to look at the "manual" for each of the commands listed above with the command `man` to learn more about them.

Note: For the `less` command, you enter a separate "window" within the terminal to view the contents of a file. In order to quit the window and stop viewing the file, you can press the `q` key.

How can we copy files?

We can copy things in the terminal using the `cp` command.

Say we wanted to copy our test file we made, `test_file.txt`, to another file called `another_test_file.txt`. We can do that with the command:

In [2]:
```
# List all of the files available in the current directory
echo "All of the files available before making a copy of test_file.txt:"
ls

# Copy the contents of test_file.txt to a new file called another_test_file.txt
cp test_file.txt another_test_file.txt

# List all of the files available in the current directory
echo ""
echo "All of the files available after making a copy of test_file.txt:"
ls

# Show the contents of the new file we made
echo ""
echo "Contents of another_test_file.txt:"
head another_test_file.txt
```

```
All of the files available before making a copy of test_file.txt:
test_file.txt

All of the files available after making a copy of test_file.txt:
another_test_file.txt test_file.txt

Contents of another_test_file.txt:
Hello world!
```

We now have two identical copies of the same file in the same directory!

How can we move a file?

If we want to move a file, we can "move it" with the `mv` command.

Let's make a new folder called `new_location` then move our `another_test_file.txt` into that folder:

In [3]:
```
# Create a new folder called new_location
mkdir new_location

# Move our file to that folder
mv another_test_file.txt new_location/

# View the contents of the new folder and the original location
echo "View the contents of the new_location/ folder:"
ls new_location/

echo ""
echo "View the contents of the current folder we're in:"
ls
```

```
View the contents of the new_location/ folder:
another_test_file.txt

View the contents of the current folder we're in:
new_location  test_file.txt
```

From this, we can see that we were able to successfully move the file, `another_test_file.txt` , to the new folder we created within `new_directory_1` called `new_location` .

Note: We can also use the `mv` command to move folders to new locations.

Note: The `mv` command also works for renaming files and folders. For instance, if we wanted to rename `test_file.txt` to `revised_test_file.txt` , we would use the command: `mv test_file.txt revised_test_file.txt` . The contents of the file would be the same, but its name would now be different.

How can we delete files?

We can use the command `rm` to "remove" files from our computer.

Warning: Unlike deleting files on your desktop, **there is no trash folder for files and folders that you delete**!
You need to be absolutely sure that you are prepared to lose the file or directory that you delete with the `rm` command as it will be impossible to recover after deletion!

Here's more information on this topic: "Where do files go when the rm command is issued?" post on StackOverflow (https://unix.stackexchange.com/questions/10883/where-do-files-go-when-the-rm-command-is-issued).

All you need to do is make sure you are in the directory that houses the file you wish to delete then perform the following:

```
In [4]: # Say we want to delete the file test_file.txt, we can delete it with the rm command

        # View the current directory we're in, as well as the contents
        pwd
        echo "Contents of the directory before removing test_file.txt:"
        ls

        # Afterwards, delete the test_file.txt file
        rm test_file.txt

        # Look at the contents of the directory again after removing the file
        echo ""
        echo "Contents of the directory after removing test_file.txt:"
        ls
```

```
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_di
rectory_1
Contents of the directory before removing test_file.txt:
new_location  test_file.txt

Contents of the directory after removing test_file.txt:
new_location
```

With this command, we were able to successfully remove the `test_file.txt` !

Removing a Directory

To remove a directory, we can still use the `rm` command, but we need to include the `-r` flag to "recusively" remove the directory and all subdirectories within it.

Let's remove the `new_location` folder and all of its contents:

```
In [5]: # Remove the new_location folder
        rm -r new_location/

        # Look at the contents of the directory after removing the new_location folder
        ls
```

This folder is now empty!

Removing an Empty Directory

If a directory is completely empty, we can remove it using the `rmdir` command instead of `rm -r` ! Let's remove the current folder that we're in because it's empty:

In [6]:
```
# View the current working directory before changing into one level above it using the
 .. notation
pwd
cd ..
pwd

# List all of the directories present before deleting,
# remove the new_directory_1 folder,
# then list all of the directories present after deleting this folder
echo "Contents of the directory before removing new_directory_1:"
ls

# Afterwards, delete the test_file.txt file
rmdir new_directory_1

# Look at the contents of the directory again after removing the folder
echo ""
echo "Contents of the directory after removing new_directory_1:"
ls
```

```
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal/new_di
rectory_1
/Users/mragsac/Dropbox/School/BISB-Bootcamp-2020/day1/module1a_bench-to-terminal
Contents of the directory before removing new_directory_1:
00_Bench-To-Terminal_Presentation.pdf
01_Bench-To-Terminal_Basic-UNIX-Commands_Traversing-Creating-New-Directories.ipynb
02_Bench-To-Terminal_Basic-UNIX-Commands_File-Folder-Editing-Viewing.ipynb
README.md
media
new_directory_1
new_directory_2

Contents of the directory after removing new_directory_1:
00_Bench-To-Terminal_Presentation.pdf
01_Bench-To-Terminal_Basic-UNIX-Commands_Traversing-Creating-New-Directories.ipynb
02_Bench-To-Terminal_Basic-UNIX-Commands_File-Folder-Editing-Viewing.ipynb
README.md
media
new_directory_2
```