**Project 1: Familiarity with UNIX/Linux**

Atuhaire Ambala and Ricardo Escarcega

Prof. Isac Artzi

CST – 315

May 12th 2024

**Part 1**

1. Five features and what happens when you access them:
2. Listing Files ($ls): This feature has to have some kind of access to long term memory of some sort as files are stored long term. After reading these files from this location then it is retrieved and displayed in the order the user might specify (date modified, size, name, etc.,).

A computer screen with green text

Description automatically generated

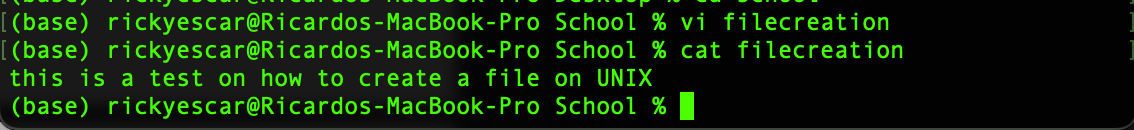
1. Login: Unix has the feature of being able to login to a profile and initialize a session using that user. Logging in I would assume would involve the kernel being able to look through it’s already stored user profiles and hashed passwords and checking to see if they match the current user that is attempting to login. Once it does matches then it would retrieve all of its history from memory.A screenshot of a computer screen

   Description automatically generated
2. Logout: Unix also has the feature of being able to logout a user from its session and break all connections. This feature must have some access to being able to wipe a session clear using the shell to process the request but then having the kernel in some way or another delete/erase any instance of the user being there while saving it simultaneously to memory so when the user logins again it will be able to start where it left off.

A computer screen with green text

Description automatically generated

1. Creating Files ($vi): Depending on the file that is being created would also depend how much access or depth is allowed to the user when creating the file. Creating a regular file would probably just be to have the shell receive the command and then have the kernel check to see if the file already exists, if there is space, and if the user has the access to create said file. Once these things check off then the shell would create the file with the given file name and contents.A screen shot of a computer

   Description automatically generated
2. Editing Files ($vi filename): Editing files starts off with being able to access the file that you want to edit first. The shell would process the command and determine whether you have the authority to access the file. The kernel then would retrieve the file from memory where then you will make your edits and it would record back to memory once edited and saved.

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Here is a screenshot from my terminal where we can see that I’ve navigated to where I want to create the Directory CST-315. Upon creating this directory I checked again to make sure it was there and it indeed says it exists.

Steps:

1. ls (To see a list of files)
2. cd (till I have navigated to the location I want to create CST-315)
3. mkdir CST-315 (create directory CST-315)
4. cd CST-315 (enter CST-315 directory)
5. git init (initialize directory as Git reprository)

A computer screen with green text

Description automatically generated

1. Created a README file using echo and saving it to the directory

A black screen with green text

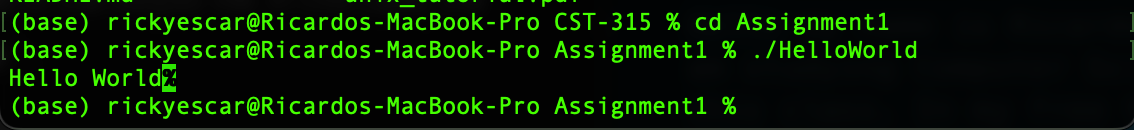
Description automatically generated

A screenshot of a computer

Description automatically generated

1. Screenshot showing the directory CST-315 containing a directory inside of it Assignment1 where a HelloWorld.c file livesA screenshot of a computer

   Description automatically generated
2. Once compiled without errors the HelloWorld.c can displayed to the console using ./HelloWorld



**Part 2**

**Repo Link: https://github.com/AtuAmbala/CST-315**

**Screenshots of commands**

1. whoami

Display who is currently using the shell

A black screen with white text

Description automatically generated

1. users

display the users of the shell

A screen shot of a computer

Description automatically generated

1. Ls

-l

(long listing format): This option displays detailed information about files, including permissions, number of links, owner, group, size, and modification date.

A black background with white text

Description automatically generated

-a

(all files): This option lists all files in the specified directory, including hidden files (those starting with a dot **.**).

A black background with white text

Description automatically generated

-lh

(human-readable): This option prints file sizes in a human-readable format, such as "K" for kilobytes or "M" for megabytes.

A black background with white text

Description automatically generated

1. chmod

A screenshot of a computer program

Description automatically generated

Here, **7** sets read (4) + write (2) + execute (1) permissions for the file owner.

A screen shot of a computer

Description automatically generated

Users in the same group, are assigned read and write permissions.

A screen shot of a computer

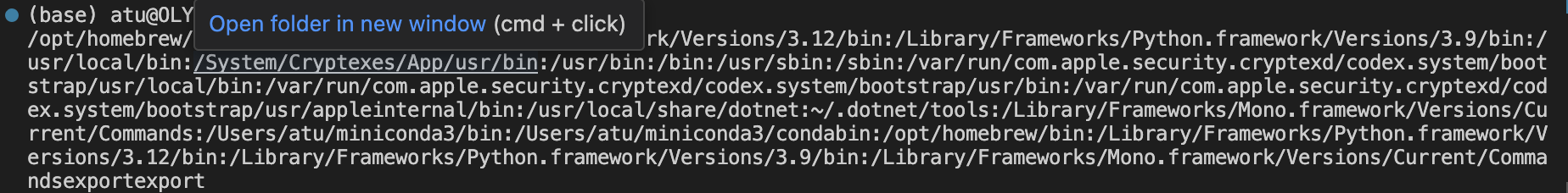
Description automatically generated

Gives the owner full access and others get execute access.

Using the numbers overwrites the previous permissions while the plus sign adds permissions.

1. Check the value of any 3 environment variables

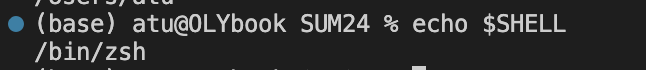
PATH : The **PATH** variable contains a colon-separated list of directories where executable files are located. It is used by the shell to find the location of commands.



HOME: The **HOME** variable stores the path to the current user's home directory.



SHELL: The **SHELL** variable contains the path to the current shell executable.



1. Using grep

To use **grep** to search for specific content in a file

A black screen with white text

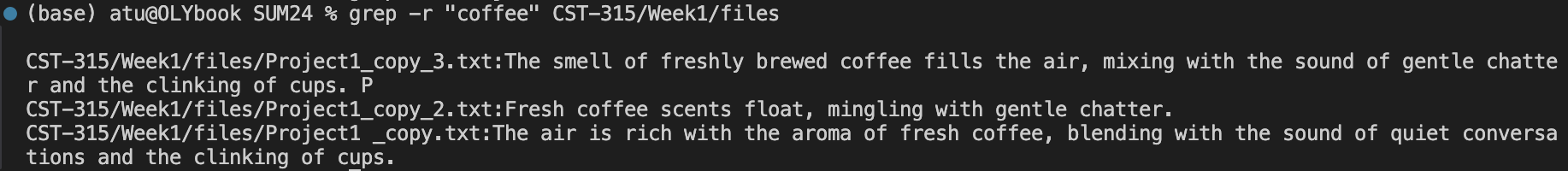
Description automatically generated

By default, **grep** will display the lines in the file that contain the word. If you want **grep** to ignore case, you can use the **-i** option:

A black background with white text

Description automatically generated

1. Use *sort, grep*, and *piping* to find a string in a list of files in a directory.



“grep -r 'search\_string' /path/to/directory” searches for the 'search\_string' in all files in the specified directory (/path/to/directory). The -r option tells grep to search recursively through subdirectories.

sort sorts the output of grep alphabetically.

When using the **grep** command in the terminal, it's important to use straight quotes (**"**) instead of curly quotes (**“”**).

grep -r “coffee” CST-315/Week1/files

1. The super user account

The super user account, often referred to as the "root" account in Unix-like operating systems, is a special user account with unrestricted access to all files and commands on a system. This account has the highest level of privilege and can perform any operation, including modifying system files, installing software, and managing user accounts.

The root account is typically used for administrative tasks that require elevated privileges beyond what a regular user account has. Some common scenarios where the root account would be used include:

1. Installing Software: Many software installations require root privileges to write to system directories or modify system configuration files.

2. System Configuration: Changing system-wide settings or configuring services often requires root access.

3. Managing Users: Creating, modifying, or deleting user accounts usually requires root privileges.

4. System Maintenance: Performing system backups, updates, and maintenance tasks often requires root access.

5. Troubleshooting: Diagnosing and fixing system issues sometimes requires root access to view system logs or modify system settings.

It's important to note that using the root account comes with risks, as any mistake or malicious command run as root can have serious consequences, including damage to the system or loss of data. As such, it's recommended to use the root account sparingly and only for tasks that require elevated privileges. Many Unix-like systems also provide tools, such as `sudo`, to allow regular users to perform specific commands with root privileges, providing a safer alternative to using the root account directly.

A black and white text

Description automatically generated

Atu had some issue running logging into Super user.

1. Five additional distinct, useful commands

**du**: Displays disk usage statistics for files and directories.

A screen shot of a computer

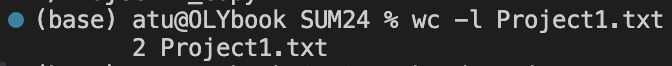
Description automatically generated

**find**: Searches for files and directories in a directory hierarchy based on a given condition.

A screenshot of a computer

Description automatically generated

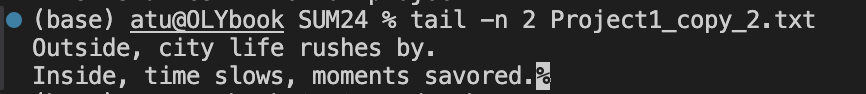
**wc**: Counts the number of lines, words, and characters in files or input.



**head**: Outputs the first part of files. You can also specify the number of lines.



**tail**: Outputs the last part of files. You can also specify the number of lines.



1. Test.sh file

Code

A screenshot of a computer program

Description automatically generated

Output

A screenshot of a computer program

Description automatically generated

**References**

"Tutorials Point - Unix Tutorial." Tutorials Point, n.d., <https://www.tutorialspoint.com/unix/index.htm>.

"Introduction." Command Line Tutorial, Read the Docs, n.d., <https://command-line-tutorial.readthedocs.io/introduction.html>.

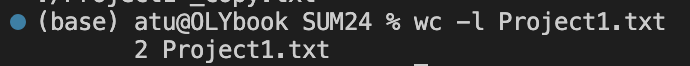
1. Five distinct useful commands

**find**: Searches for files and directories in a directory hierarchy based on a given condition.

A screen shot of a computer

Description automatically generated

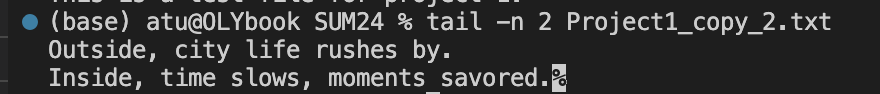
**wc**: Counts the number of lines, words, and characters in files or input.



**head**: Outputs the first part of files. You can also specific how many lines you want.



**tail**: Outputs the last part of files. You can also specify how many lines you want.



**du**: Displays disk usage statistics for files and directories.

A screen shot of a computer

Description automatically generated