**Assignment 5**

**Deliverables**

* This lab assignment is divided into 6 tasks.
* Use the same document to answer question and add screenshots.
* After completing all 6 tasks show this document to your lab professors for grading.
* Demo your work to your lab professor during lab class for grades.

(No submission will be accepted, student must demo during lab class).

**Note**: All Linux Commands are with small letters!

**Assignment’s TASKS:**

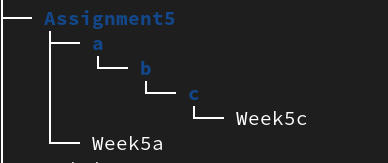
For Task 1 to 6 (Total 15 points, **you lose 1 point for each mistake**)

**Task 1**

**Task-0. Create below file structure under your personal HOME directory**

**Assignment5,a,b,c – are directories.**

**Week5c, Week5a – are files**



**Task 1.**

**Task** **1(a)**

**Create new commands (alias) in Linux:**

In this part using “alias” you will learn how to create new commands that do more for you. This is a strong feature in Linux that give you lot of power on your administration/support tasks:

* 1. Inside your HOME directory; type:

$  **ls -ail**

* 1. Explain the difference between **ls** and **ls -ail**

Ans:-

* 1. Now create an alias (new command) called “**list**” equivalent to “ls –ail”

Now execute the above alias you created

Add screenshot

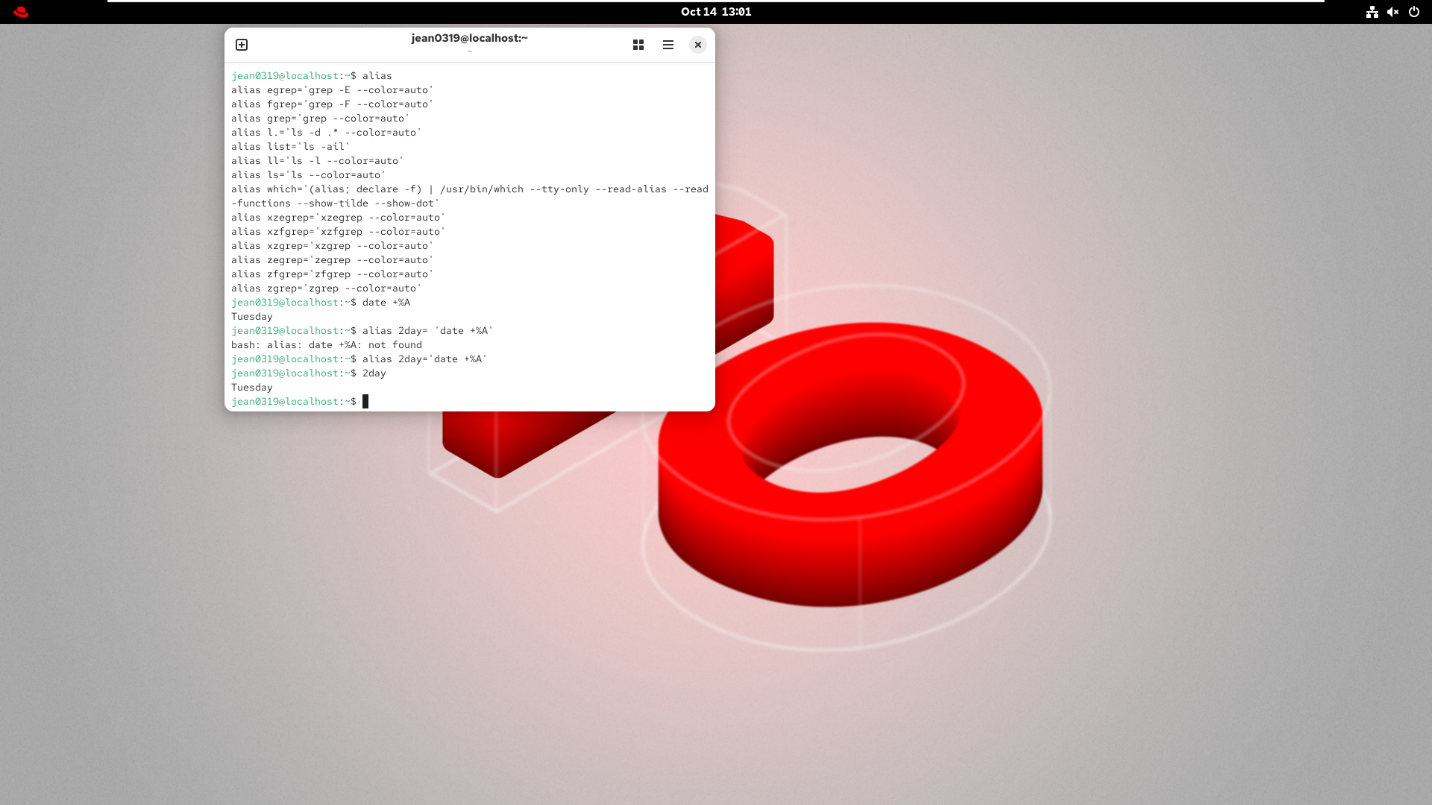
* 1. Now find what other aliases reside in your Linux box (refer notes to see command used to see all existing aliases on your OS), are you able to see list alias?

Screenshot

* 1. Now create an alias that when you type **2day** as a command output will show you weekday

For example

screenshot here that the 2day works



* 1. Now create an alias by name that when you type **files** as a command output will produce all the possible pathnames for all files under /home/<user> directory hierarchy.

screenshot of execution



**Task** **1(b)**

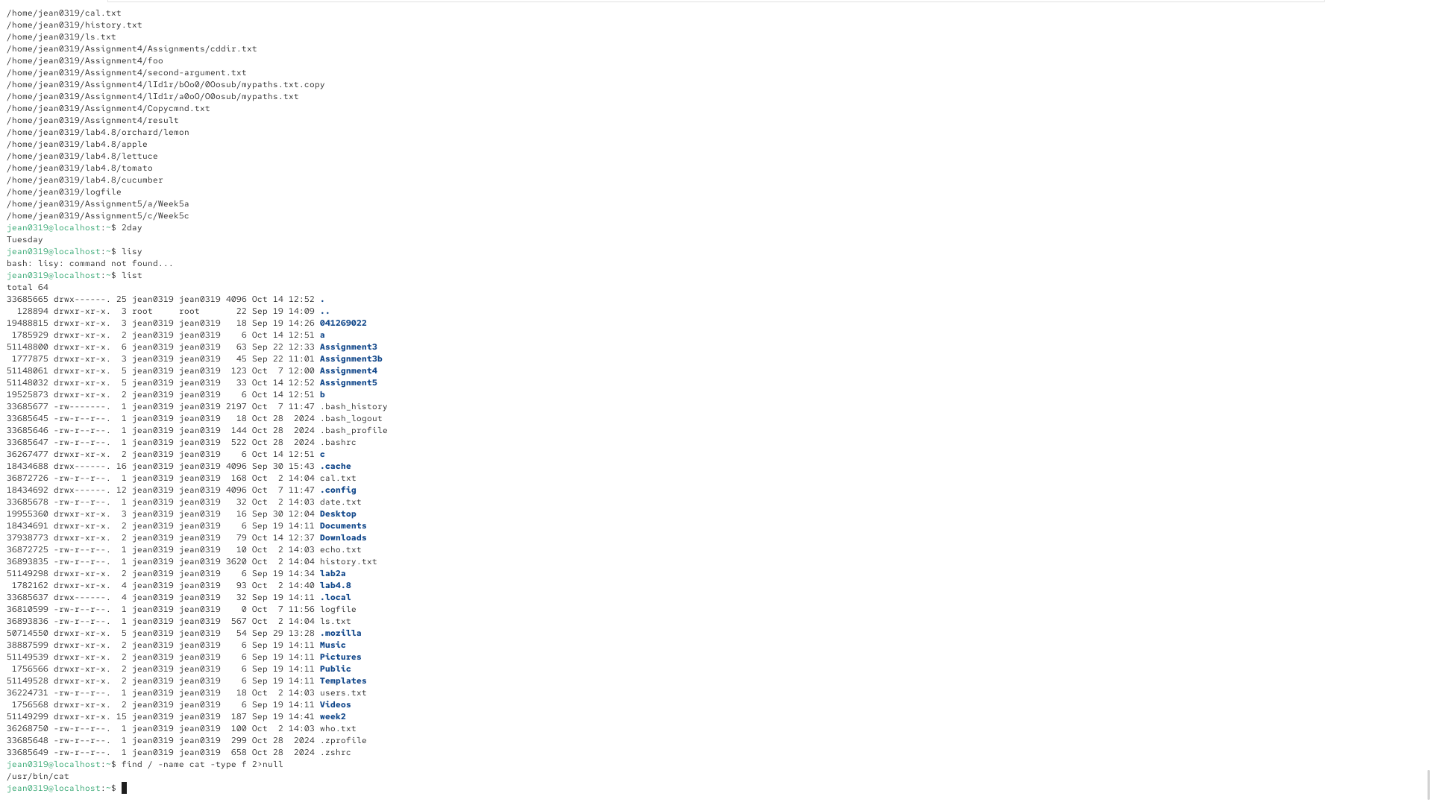
**NOTE:- Your current working directory has to be your HOME directory to complete this task.**

1. Find the location of the **cat** command Program using **find** utility.

Hint:

* Start your search from topmost directory.
* Implement error redirection if you see lot of permission deny errors, and redirect it to /dev/null
* Do some google research on file /dev/null. Now think why we redirected our errors to /dev/null

(Add your screenshots to your answers)



**Task 1(c)**

1. Find out all the files that start with **Week** and has only 6 characters in their name
2. Are you able to see the absolute path of files we created in task 0 ?

(screenshots of your result here)   
A white background with blue text

AI-generated content may be incorrect.

1. Using **wc -l** command find out how many files you’ve found in step 1 (hint: we demonstrate that in lecture, [pipeing])

(screenshots of your result here)

A white background with blue text

AI-generated content may be incorrect.

**Task 2: Redirecting output:**

**cat /etc/passwd and redirect the output to file *dump.txt*** under your **Assignemnt5** directory.

Screenshot of command line

A white background with blue text

AI-generated content may be incorrect.

Now try this:

**$ ls /home/user1 and redirect the output to file *dump.txt*** located under Assignment5 directory

**$ cat dump.txt**

The previous content of dump.txt is gone explain why?

Ans:

How to append two files using >> (append)

**$ ls /etc/passwd >> dump.txt**

**$ cat dump.txt**

The previous content of dump.txt is NOT gone!

Explain the difference between “>” and “>>”

Ans

**B) Redirecting Errors:**

Try the following commands: Under your HOME directory

**$ ls W5 Assignment5**

1. Observe the output we have one error message.
2. Now create a command line to just redirect the error message to file **dirlist** under your **Assignment5** directory.

(This practice in industry used for hiding possible run-time error from users of the service)

screenshot of command line

A white background with black text

AI-generated content may be incorrect.

1. In next step try to redirect the standard output and error message to file ~/Assignment5/Both.txt

(Hint: you can accomplish this by using **2>&1** [redirect all errors to where the output goes])

screenshot of command line

A white background with black text

AI-generated content may be incorrect.

**Piping the output of a command into another command:**

1. Try this in Assignment5 directory:

1. **$ find /etc** (has recursively generated all the possible paths starting with /etc)
2. **Modify the above command line to Count how many pathnames are generated.**

Screenshot of command line and output.

A white background with black text

AI-generated content may be incorrect.

1. Here we pipe the output of above into head command to show first 5 line ( -5 options)

**$ find /etc | head -5** ( if you change -5 to -2, the output only shows first two line of output)

1. Now create a command line to generate the pathnames **leading to the files** and count how many pathnames are generated.

Screenshot of command line and output

A white background with black text

AI-generated content may be incorrect.

1. Lets say, the output you got for above (point 4) command line is **1126** (that is number of line)

Now just grab the pathnames/lines in range 50 to 100 only, include line 50 and 100(Hint: head and tail)

**Screenshot of output and command line**

**A white background with black text

AI-generated content may be incorrect.**

Now try this:

**$ ls /etc | more**

Here we pipe the output of **ls** command into **more** command which shows the output page by page

**Task 3 Variables and Array**

**Define a variable by names as specified below:**

**Name** – store your full name.

**City**- store your city name.

**Student\_Number** – store your student number.

**Username** – store your Algonquin username.

After declaring all variables

1. **Use set command to see the variable you declared above. (Refer: Hint below)**

**Screenshot of command** **line**

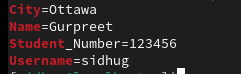
A white background with black text

AI-generated content may be incorrect.

Hint:- set command display the names and values of all shell variables and functions. Searching for specific variables in this long output is tedious.

Now recall using grep command and pipe | , implement it to create a single command line to see all the variables you defined above.

**Output should look like**



**Screenshot of output and command line**

**A white background with black text

AI-generated content may be incorrect.**

1. **Now: use the echo command and use all the variables in one command line to fetch the values of the variable.**

**Output should look like:**



**Screenshot of command line and output  
  
A white background with black text

AI-generated content may be incorrect.**

**ARRAY**

* An array is a special type of variable because it can contain more than one value.
* An array is a structured arrangement of similar data elements.
* Within shell scripting, **an array is a variable that holds multiple values**, **whether they are of the same type or different types**. It's important to note that in shell scripting, everything is treated as a string.

**Declaring an Array**

**Syntax :**

**my\_array=(value1 value2 value3)**

* my\_array is the name of array
* value1,2,3 are the multiple values store in array “my\_array”

Value1 is stored at index value 0

Value2 is stored at index value 1

Value3 is stored at index value 2

**Accessing values**

echo ${Name\_of\_array[0]}

Outputs value1 🡪 identified by index value 0 Zero”

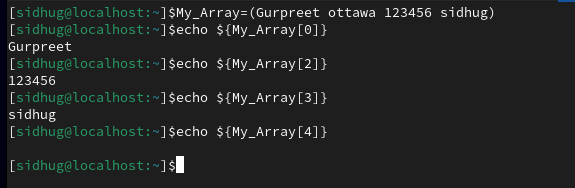
echo ${my\_array[1]}

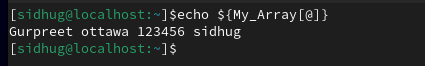
Outputs value2 🡪 identified by index value 1”

echo ${my\_array[@]}

Outputs all values stored in an array

Example in execution:





1. Now create an array by name **My\_Info** and store values Name, city, Student Number, username.

Screenshot of command line  
A white background with black text

AI-generated content may be incorrect.

1. Now fetch just the values Student number and username

Screenshot of command line and output  
  
A white background with text

AI-generated content may be incorrect.

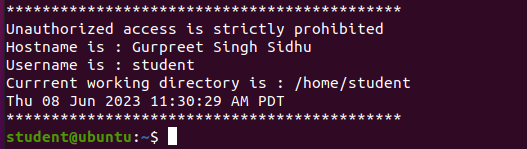
**Task 4**

**How to create a Welcome/Warning Banner**

1. You have been assigned to create a welcome **banner** for your Linux Box to show below information (Refer : Image 1 ) every time you open terminal.
2. Banner should have proper line spacing, indicating starting and ending of the banner.
3. Accomplish this task by editing **.bashrc** file located in your HOME directory (/home/student) and deploying related shell variables like: $USER, $PWD to show the below information as your
4. welcome banner using the variable combined with **echo** command.
5. Add your lines at the end of .bashrc file.

WARNING: make sure not to delete or modify the old content of this file.

* 1. **Example:** echo “Username is : $USER”(it shows the username)



**Ref: Image 1**

* 1. **For HOSTNAME variable the value is localhost (but in welcome banner I need to see your full name )**

**Hint: - You know how to manipulate the value of the variable or create your own variable.**

1. Once completed, save .bashrc, Now close and reopen terminal, you should see your welcome banner (something similar to above image1 not identical though), make sure all info are correct and then take a screenshot of the output   
   A computer screen shot of a red ring

   AI-generated content may be incorrect.

**Task 5**

**Count IP addresses used in SSH break-in attempts in January**

1. To work on this task first download the Logfile posted on Bright space under Week 5 content to your Linux machine. (Make sure you have your Linux machine connected to network
2. If your machine is not connected (talk to your lab professor)
3. Now use the find command to look where your logfile was downloaded.

Add Screen shot of your terminal

A screenshot of a computer

AI-generated content may be incorrect.

1. Once you find the path of downloaded logfile copy that file to ~/Assignment5/logs/ (you need to create a logs directory)

Add Screen shot of your terminal   
A screenshot of a computer

AI-generated content may be incorrect.

1. Use filter commands to find the unique IP addresses used in SSH break-in attempts in January and then count how many times each IP address was used.

we need to look for lines in the logfile (you downloaded) that contain both the string **'refused connect'** and the date string '**Jan** '. now we need to extract the IP address from each line and count the number of times each IP address appears.

Hint :- Below

**First grep command** selects the lines containing the text string 'refused connect' inside the Logfile. The output of this first command (only lines containing the 'refused connect' string) goes into the first pipe, not onto the screen.

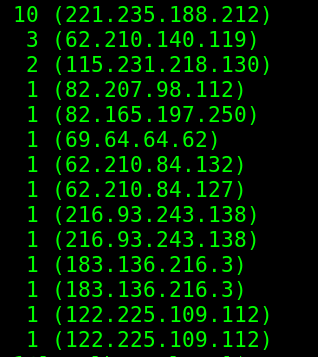
**Second grep** reads the output of the first grep from the pipe and only selects lines that also contain the date pattern for January 'Jan '. The lines being selected have to contain both the string 'refused connect' from the first grep and the string 'Jan ' from the second grep. The output of this second grep (lines containing both strings) goes into another pipe.

The **awk** command reads the selected lines from the pipe. It displays just the last field (NF) on each line, which happens to be the IP address used by the attacker. The awk output (a list of IP addresses, one per line) goes into another pipe. (The list of addresses are not in sorted order; they are in whatever order they appear in the input file.)

The **uniq -c** command reads the sorted list of IP addresses from the pipe. It counts how many adjacent addresses are the same and sends the uniq output (lines with the count followed by the IP address with that count) into another pipe.

The **sort -nr** command reads the lines with the counts and IP addresses from the pipe. It sorts numerically and in reverse (descending) order the lines containing the leading count numbers and sends the sort output (sorted lines, each containing a count and an IP address) onto the screen.

Your output after executing the command line will look like this



After confirming your work (show your output to your lab professor)

**Task 6** - **Conversion between symbolic mode and octal mode**

1. As shown in the class, here you will do conversion for each nine-character symbolic permission, give the equivalent three-digit octal permission (the red is as example) :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Symbolic Mode | Octal Mode | User/Owner | Group | Other |
| rwxrw-r-x | \_ 765 | \_ 7 | \_ 6 | \_ 5 |
| r---wx-w- | \_ | \_ | \_ | \_ |
| --x------ | \_ | \_ | \_ | \_ |

1. More of conversion between octal mode and symbolic mode.

This task is reverse of above task so for each three-digit octal permission in the following table, give the equivalent nine-character symbolic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Octal Mode | Symbolic Mode | User/Owner | Group | Other |
| 001 | \_ | \_ | \_ | \_ |
| 421 | **r - - -w- - - x** | **\_ r - -** | **-w-** | **- - x** |
| 300 | \_ | \_ | \_ | \_ |
| 504 | \_ | \_ | \_ | \_ |
| 756 | \_ | \_ | \_ | \_ |

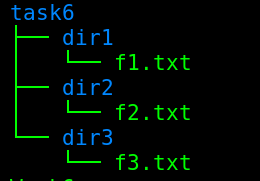
**Task 6**

**Analyze Effect of permission on File/Directory in practice**

Create below file structure under your ~/Assignment5 directory

Blue – directories

Green - files



1. Add any content to these files (content must contain its corresponding filename).
2. Open all the files (f1.txt, f2.txt, and f3.txt) Together (need to see corresponding file name as well in output)

add a screenshot of the output

Now just change the permissions for owner set , don’t change the permissions for group and other set as specified below

1. Give execute permission to f1.txt, write permission to f2.txt and read permission to f3.txt

(f1.txt1 should have only execute permission, f2.txt only write permission, and f3.txt only read permission).

1. Execute the command to see the permissions of all files.

Add screenshot

1. Now open all files together, you are not able to see the content of which file? why?

Ans:-

1. Now try to append any text to all three files (using >>).
2. For which file you were able to append text and why not to other files?

Ans:-

Add screenshot of command lines

**Changing permissions for dir1,dir2 and dir3**

Just change the permissions for owner set, don’t change the permissions for group and other set.

1. Give only execute permission to dir1, write permission to dir2 and read permission to dir3.

(dir1 should have only execute permission, dir2 only write permission, and dir3 only read permission).

1. Execute the command to see the permissions of dir1, dir2, and dir3.

Add Screenshot

1. **Make task6 directory your working directory** and cd to dir1, dir2, and dir3 one by one

Explain why you were able to cd to dir1 but not for dir2 and dir3

Ans:-

1. **Make task6 directory your working directory** and Now execute ls -l command for dir1 dir2 and dir3
2. For which directory you are able to long list the content?

Ans:-

1. **Make task6 directory your working directory** and Try to see the content of files f1.txt f2.txt f3.txt using cat command.

You wont be able to see the content of any of the files.

1. Now give min permissions that are required to read the content of files f1.txt f2.txt f3.txt.

**Fill in the table below with min permissions you assigned to dir1,dir2,dir3, f1.txt, f2.txt, f3.txt**

|  |  |
| --- | --- |
|  | Permissions |
| dir1 |  |
| dir2 |  |
| dir3 |  |
| f1.txt |  |
| f2.txt |  |
| f3.txt |  |

**After completing your assignment show your document to your lab professor**