**Linux Tasks**

# Task # 1) Create a new directory

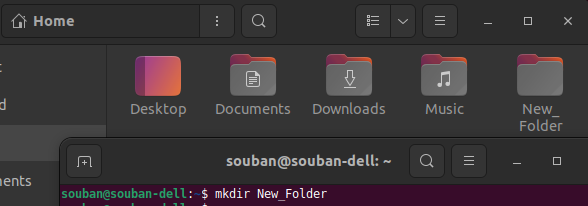
**Syntax:**

**mkdir directory\_name /create directory in the present path**

**mkdir ~/path/directory\_name /create directory in any path from anywhere**

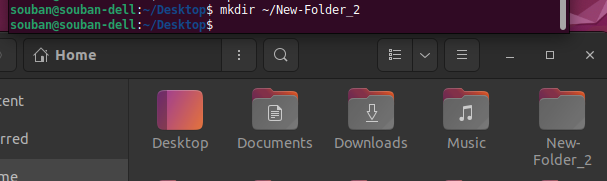
1. Creating a new dIrectory in Home **‘New\_Folder’**

**Explanation:** for creating a new directory/folder you use mkdir (make directory) command. It will create the directory in the path you are present in. RIght now the directory is created in the home directory. If we want to create another directory in it, we will first go to another path and use this command again. However there are other ways as well to do this.



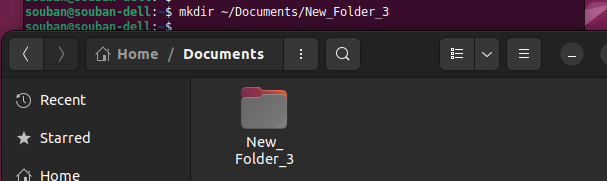
1. Creating a new directory **‘New\_Folder\_2’** in **Home** being in the **Desktop.**

**Explanation:** Now I have changed my directory/path to **Desktop** but I want to create a new directory to home. I will use the command **mkdir ~/New-Folder\_2** and it will create the new directory/folder at home.

****

1. Creating a new dIrectory **‘New\_Folder\_3’** in **Documents Folder** being in the **Desktop.**

**Explanation:** We can also create a new directory inside another directory while being on some other directory. For example, we can create **New\_Folder\_3** inside the **Documents** folder while being in the home directory. The main syntax this depends on the command **mkdir** while using the sign **tilde ~** and mentioning the path using **slashes /**.

****

**Note:** This was done in an explanatory way for my practice.

# Task # 2) Change to a different directory

**Syntax:**

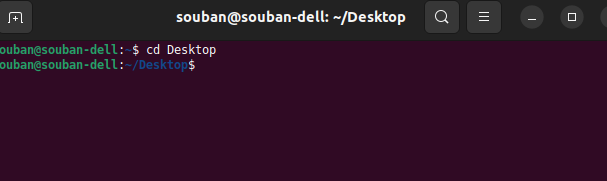
**cd /go to home directory**

**cd directory\_name** or **path/directory\_name /go to specified directory from home**

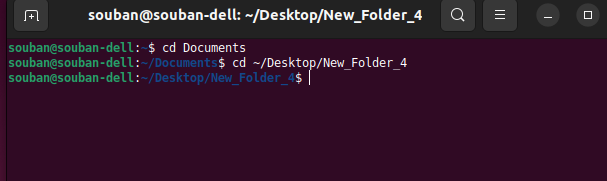
**cd ~/path/directory\_name /go to specific directories from anywhere**

**Explanation**

If we want to change our directory to some specified directory from **home**, let say from home to **Desktop** and if we want to go to a specific directory we will mention a path.

****

If we want to change our directory directly to some other directory. We will add **‘~/’** before the path we mention. Such as in the example below. We moved from **Documents** to **New\_Folder\_4** in **Desktop**

****

# Task # 3) List the contents of a directory

**Syntax:**

**ls /Shows all the contents files+folders without any arguments**

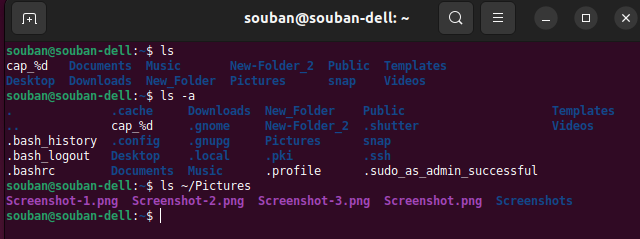
**ls -a /Shows all the contents files+folders without any arguments**

**ls ~/directory\_name /Shows all the contents files+folders without any arguments**

**Explanation**

**Ls** command is used for listing the contents of any directory unless the directory path is provided. In the first, I enlisted the home directory, in the second place I enlisted all the contents along with hidden contents as well. There are plenty of other arguments which can be taken out from –help.

Lastly we can also enlist the contents of any directory by providing the directory name and path.

****

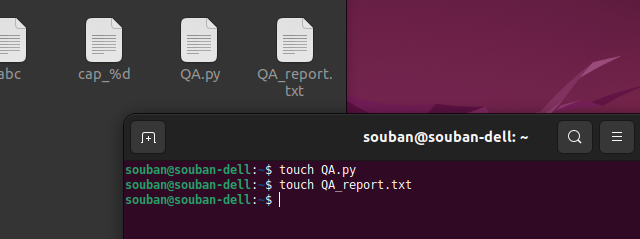
# Task # 4) Create a new file

**Syntax:**

**touch filename.(file type) /creates a file with specified file format**

**Explanation**

For creating file of whatever type or format we want we use touch commands along with filename and its type. If we don't mention file type. The terminal will create a text file in default.

****

# Task # 5) Open a file in Text editor

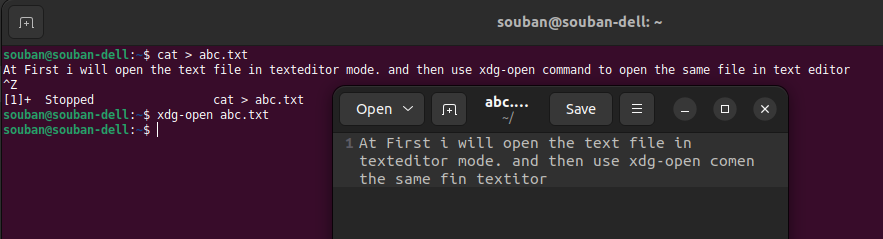
**Syntax:**

**cat > filename.(filetype) /open file in text editor in terminal**

**xdg-open filename.(filetype) /open the file in text editor application**

**Explanation**

We can open the file in the terminal as a text editor by using the first command but if we want to open the file in text editor application **xdg-open filename.(filetype)**.

****

# Task # 6) Copy a file to a new location

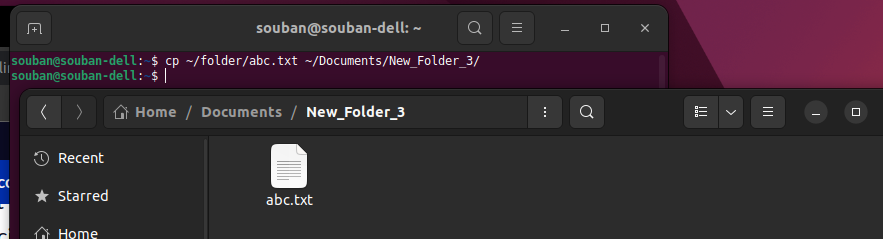
**Syntax:**

**cp [~/path/filename.(filetype)] [~/path] /to copy the file to any destination from any source folder**

**cp -r [source] [destination] /copied the file from the specified source and putting it in the specified directory. If the directory is not present it will create it**

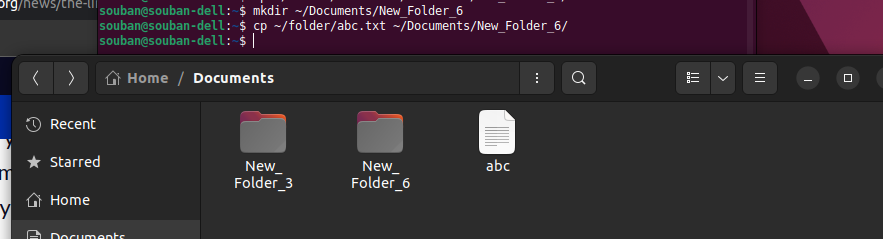
**Explanation**

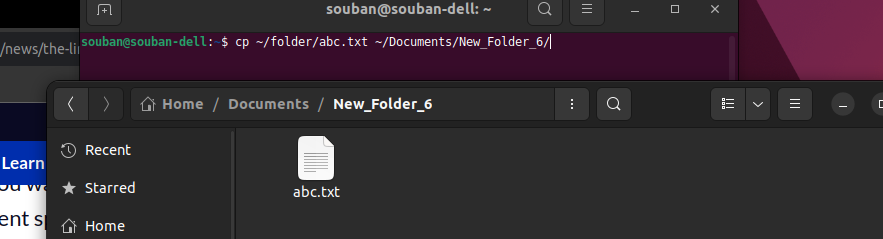
By using the above command we can copy the file from the source folder/directory to any other location/directory as far as we specify the path to it. In my command I am copying the file **abc.txt** from the directory folder located in **home** to the directory of **Documents** inside the folder named **New\_Folder\_3**.

****

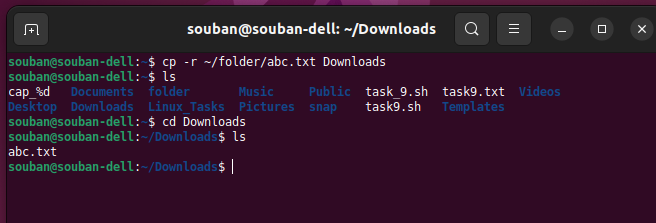
Similarly we can copy any file to any new directory by first creating the directory/location using the mkdir command and then copying the file to that location by using another command of cp as discussed above.

Just like in the steps below, making a new directory as **New\_Folder\_6** in the **Documents** folder and later by using the command of cp I am copying the file abc.txt into it which is shown in the next figure.





Cp -r command is used to copy the file abc.txt from the source folder and then moving it to new location Downloads.



# 

# Task # 7) Rename a file

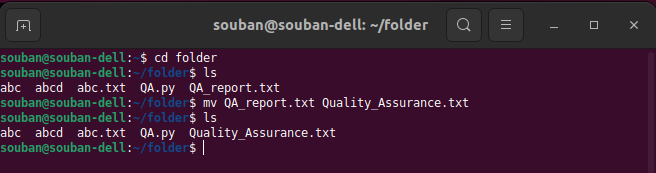
**Syntax:**

**mv [file1.(type)] [newfilename.(type)] /rename the file**

**Explanation**

By using the mv command and writing the name for the original file and then the new name for the file we can rename the file.

In the figure below, I have listed the files of the directory named **folder** and then renaming the original file named **QA.txt** into Quality\_Assurance.txt.

****

# Task # 8) Delete a file

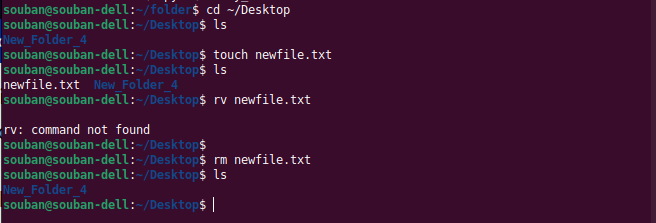
**Syntax:**

**rm [filename.(type)] /remove file**

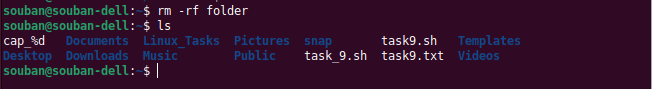
**rm -rf [directory\_name] /remove the directory**

**Explanation**

Going to the desktop directory and listing all the files and folders. Creation of new file named as **newfile.txt** and then removing it using the rm command and displaying the contents of the directory again by listing it.

****

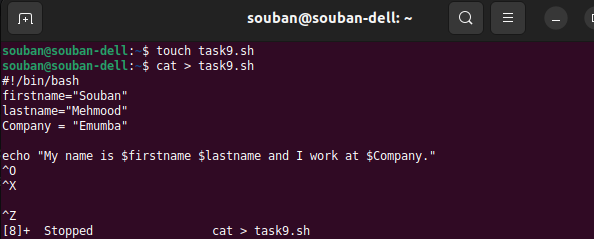
Removing the whole directory from the path.

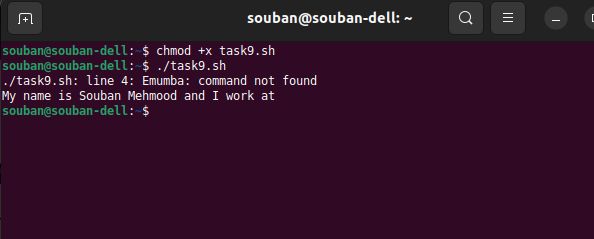
****

# Mv Task # 9) Use variables to store and output data

**Explanation**

This task is done in a .sh file by creating it using touch command and opening it with cat command. The variables are made termed as firstname, last name and company. An echo command which is similar to ‘cout’ in c++ and print as well. This echo is used to display the variables but with every variable in the echo statement we have to use $ sign.

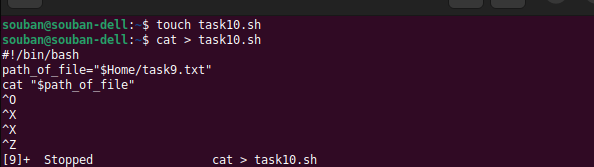
****

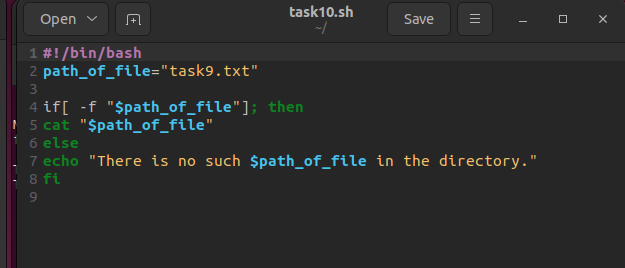
****

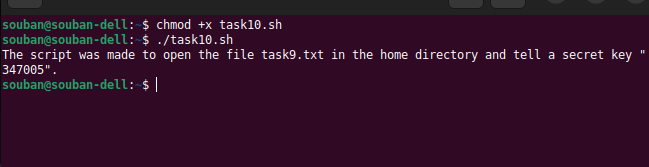
# Task # 10) Create a simple script to automate a task

**Explanation**

By touch command the file is created then by cat > command the file editor is opened and a simple task of opening the task9.txt file is done using the cat command on the filename mentioned inside the variable path\_of\_file.

****

****

****

# Task # 11) Use if statement to make decisions in a script

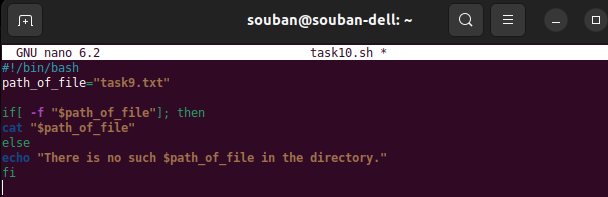
**Syntax:**

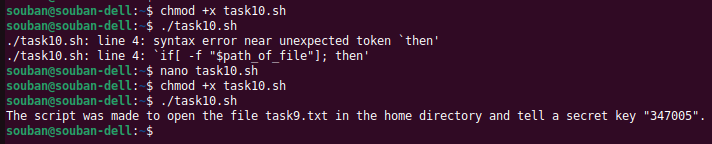
**cd /go to home directory**

**Explanation**

This script is opening the text file by reading the name of file it is specified. First the if statement checks the correct file and as soon it is assessed correctly it will display the contents inside the file using cat command.

****

****

****

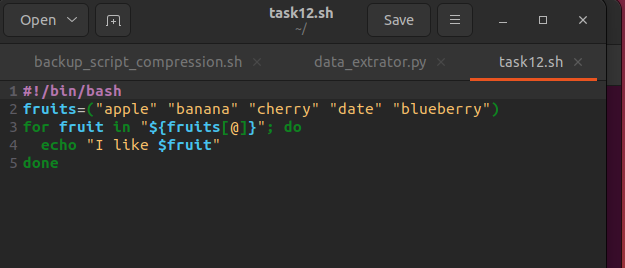
# Task # 12) Use loops to iterate over a set of data

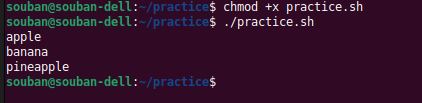
**Syntax:**

**Explanation**

Creating the .sh file using nano and editing the file, declaring the string of fruits as per the figure below. For loop is written which will return the variable fruits array along with the string ‘i like’. For the length of array the iteration will run.

****

****

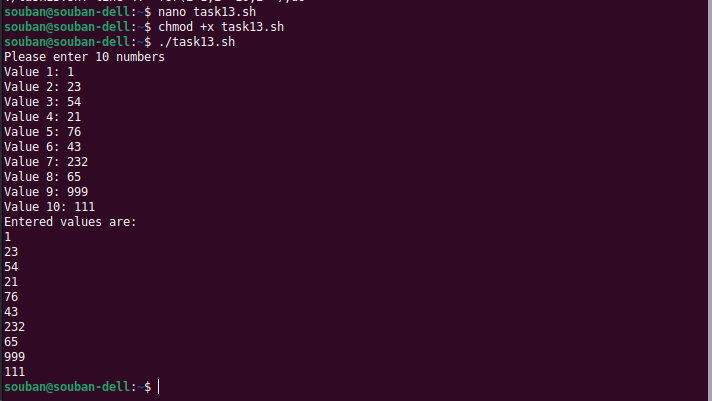
****

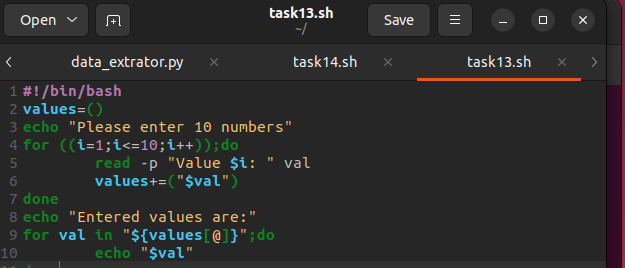
# 

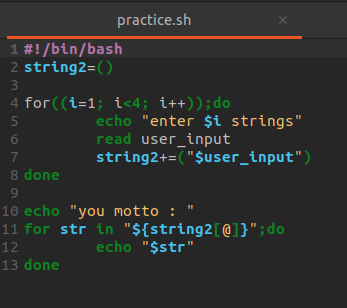
# Task # 13) Read input from the user

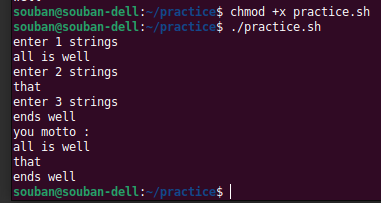
**Explanation**

Declaring the value argument for collecting the values by user. Displaying the message for the user to enter the 10 numbers. For loop for 10 iteration where it is taking the user input from the terminal inside the i variable. Once it is done 10 times, the values are displayed on the screen with a for loop by using the argument values

****

****

****

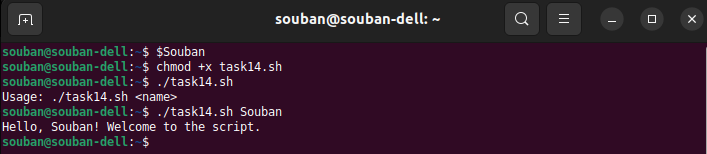
****

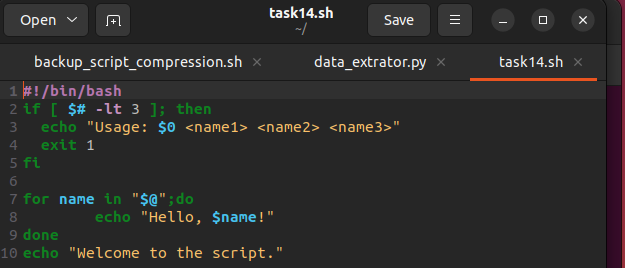
# 

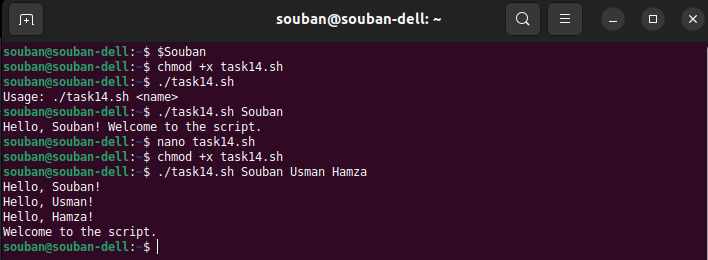
# 

# Task # 14) Use command line arguments to customize a script

**Explanation**

****

****

****

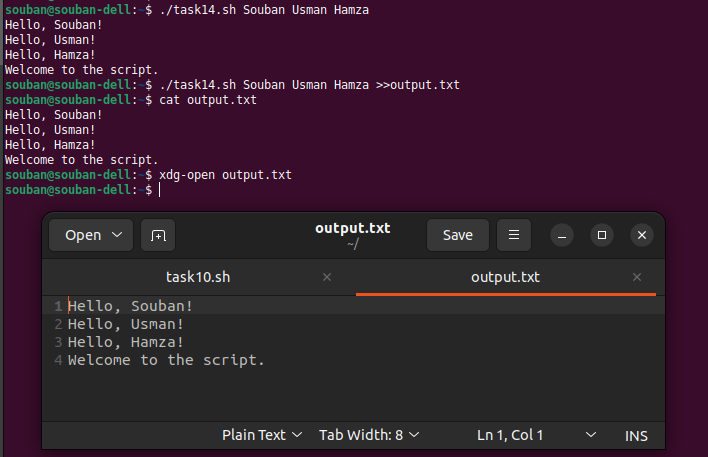
# Task # 15) Redirect output to a file

**Syntax:**

**[object ] >> [defined output]**

**Explanation**

I am executing and running the script of task 14 and then displaying the output contents by first storing them inside a text file and later in the next command line i am displaying the text file.

****

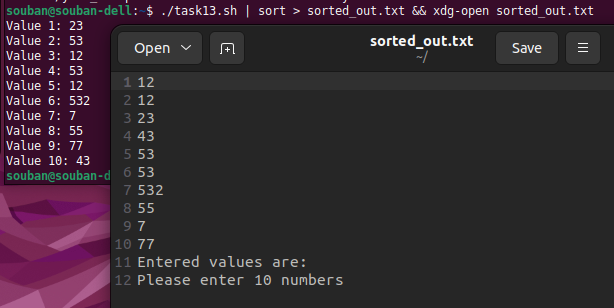
# Task # 16) Chain commands together using pipes

**Syntax:**

**| and &&**

**Explanation**

These are basically the logic operators ‘OR’ & ‘AND’. By OR means ‘it has to do this along with this’ and ‘AND’ means it has to do this and this as well’. which means in the command displayed below it is running the script along with sorting the output into a text file and lastly the previous commands are completed so it is opening the sorted command in the text editor.

****

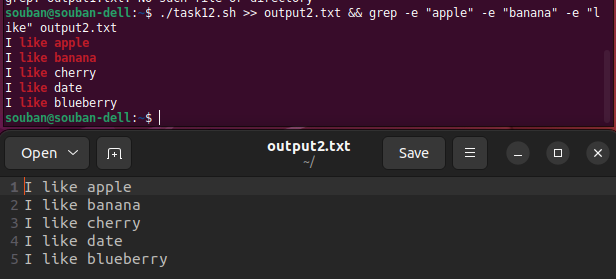
# Task # 17) Use grep to search for text within a file

**Syntax:**

**Grep -e “word”**

**Explanation**

First i ran the old script file of the task which was generating some string lines. Next the output of the file is stored in a text file named output.txt and then chained these command with grep which is used for searching the text withing a file and highlighting on the console. So i used multiple texts like ‘apple’, ‘banana’ and ‘like’.

****

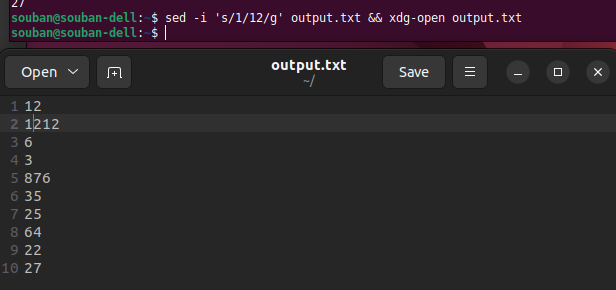
# Task # 18) Use sep to replace text within a file

**Syntax:**

**Sed -i ‘s/word/word/g’ filename.type**

**Explanation**

-i in this is used for case insensitive search and the search objects and replacements are 1 into the integer 12. Also I chained the command with xdg-open to open the file with replacements.

****

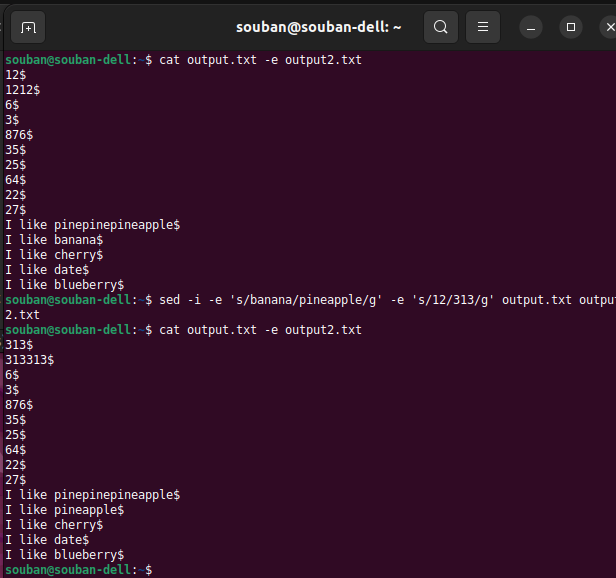
# Task # 19) Find and replace text in multiple files

**Syntax:**

**sed -i -e ‘s/word1/word2/g’ -e ‘s/word1/word2/g’ file1.type file2.type**

**Explanation**

In the figure below it is first the display of all the contents inside the two text files and then in the next we used sed command along with argument -i that makes it insensitive search and -e for separate script or command. Then mentioning the word replacings in each segments along with the same order is mentioned the files needed to be accessed and replaced.

****

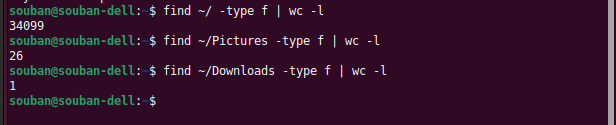
# Task # 20) Count the number of files in a directory and its subdirectories

**Syntax:**

**find ~/path -type f |wc -l /finding the file and counting the files and directories inside the path**

**Explanation**

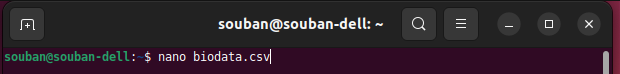
I am chaining the commands of find and count. We first write the command for finding the directory which is given. Then write the type -f for regula search and then after chaining the command with another one which is wc -l which will count the number of files.

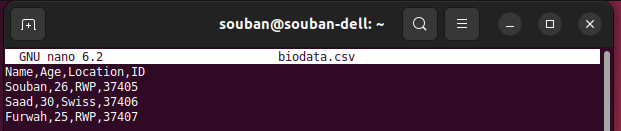
****

# Task # 21) Parse a CSV file and extract data

**Explanation**

By using the nano command we create the csv file with the name biodata. Then we enter the data inside it with Name, Age, Location, ID as shown below. Now after saving the file we have to create a python script for scraping the data from the file.

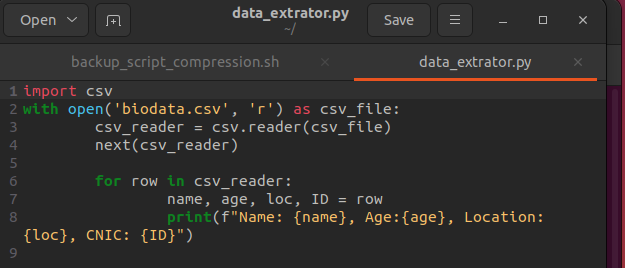
****

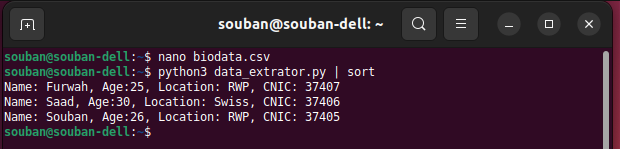
****

For the python file, we create the file using nano and start writing the script. Initially we need to import the library for importing csv. Now recursively we need to open the csv file and load the file. For that a syntax of with open() is used and the filename is inserted.

Next the python engine has to read the data from the the csv file using the csv.reader() command. And for skipping the first line we use the command ‘next’

Now a for loop in a row format is ran with printing sequence of Name, Age, Location and CNIC.

****

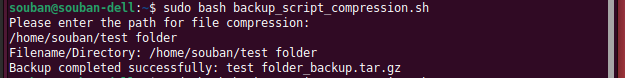
****

# Task # 22) Create a backup script that compresses files

**Explanation**

At first I created the .sh file through touch command. Now the script inside it works iin such a manner that a variable is created with the name “backup\_storage” which is storing the compressed file in home location.

Now a while statement is originated with a statement to check if the path entered by the user through read command is accurate or not. If it's not correct, the console will print the message to enter the path again. Once the path is corrected the a command for backup\_name of the file. Afterwards a directory for storage of the file is created and by using tar command the backup file which is created is stored in the backup storage path..

****

**#!/bin/bash**

**backup\_storage=~/ # Store backups in the home directory**

**# Taking user input for the directory/file to backup**

**while true; do**

**echo "Please enter the path for file compression:"**

**read -r backup\_dir**

**# Checking if the file or directory is valid**

**if [ -e "$backup\_dir" ]; then**

**echo "Filename/Directory: $backup\_dir"**

**break**

**else**

**echo "File or directory is not valid or the path is incorrect."**

**fi**

**done**

**# Creating a backup filename**

**backup\_name=$(basename "$backup\_dir")**

**# Directory for storage/backup of the file**

**backup\_filename="${backup\_name}\_backup.tar.gz"**

**# Create the compressed backup archive**

**tar -czf "$backup\_storage/$backup\_filename" -C "$backup\_dir" .**

**# Check if the backup was successful**

**if [ $? -eq 0 ]; then**

**echo "Backup completed successfully: $backup\_filename"**

**else**

**echo "Backup failed."**

**fi**