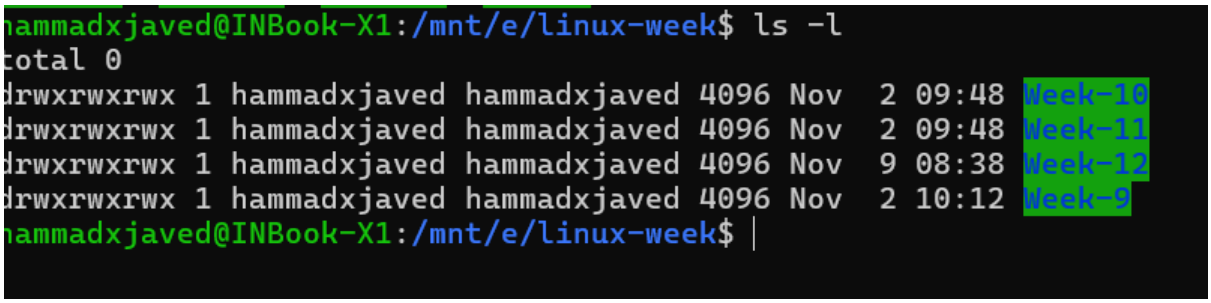
Week 8

1. Display detailed information about files using ls -l and other ls options.

Display Detailed Information About Files:

- **ls -l**: Lists files in the current directory with detailed information.

ls -l

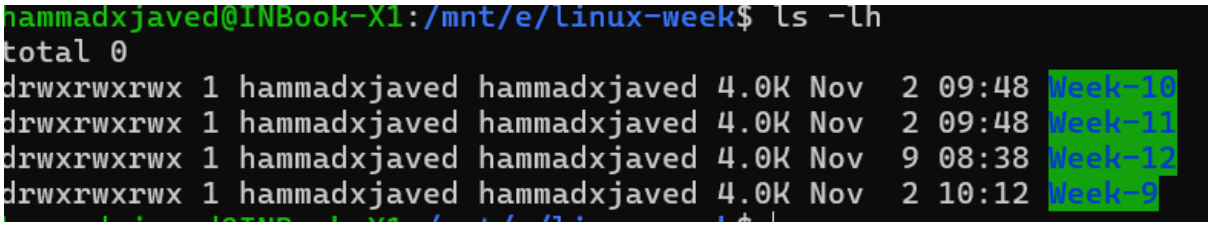


- **ls -a**: Lists all files, including hidden files.

ls -a

- **ls -lh**: Lists files with human-readable file sizes.

ls -lh



2. Change file ownership using chown and group ownership using chgrp.

Change File Ownership:

- **chown**: Changes the owner of a file.

chown new_owner file_name

- **chgrp**: Changes the group ownership of a file.

chgrp new_group file_name

3. Modify file and directory permissions using chmod.

Modify File and Directory Permissions:

- **chmod**: Changes the permissions of a file or directory.

chmod permissions file_name

For example, to give the owner read, write, and execute permissions, and others read-only permissions:

chmod 755 file_name

4. Demonstrate how to set and remove directory permissions.

Set and Remove Directory Permissions:

- To set directory permissions:

chmod permissions directory_name

For example, to give the owner read, write, and execute permissions, and others read and execute permissions:

chmod 755 directory_name

- To remove directory permissions:

chmod permissions directory_name

For example, to remove write permissions for others:

chmod o-w directory_name

5. Write a python program that reads a string which contains English alphabets and numbers. The program should create two lists L1 and L2, where L1 includes all the numbers present in the string while L2 includes all the alphabets of the string.

```
def separate_numbers_and_alphabets(input_string):
    L1 = []
    L2 = []

    for char in input_string:
        if char.isdigit():
            L1.append(char)
        elif char.isalpha():
            L2.append(char)

    return L1, L2

input_string = input("Enter a string containing alphabets and numbers: ")
numbers, alphabets = separate_numbers_and_alphabets(input_string)
print("List of Numbers (L1):", numbers)
print("List of Alphabets (L2):", alphabets)
```

```
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Min
i Project)\Weeks\MCA-III_LAB> & C:/Users/Hammad/AppData/Local/Microsoft/WindowsApps/python3.12.ex
e "c:/Users/Hammad/OneDrive - myamu.ac.in/Desktop/MCA/MCA III/CAMS3P01 Laboratory Course-III (Min
i Project)/Weeks/MCA-III_LAB/Week-8/Q5.py"
Enter a string containing alphabets and numbers: bsdjh3q32wavinje
List of Numbers (L1): ['3', '3', '2']
List of Alphabets (L2): ['b', 's', 'd', 'j', 'h', 'q', 'w', 'a', 'v', 'i', 'n', 'j', 'e']
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Min
i Project)\Weeks\MCA-III_LAB>
```

6. Write a program in python to find vowels having maximum number of instances in a given file. (Note: File contains variety of data such as English alphabets, numbers etc.).

```
def find_max_vowel_in_file(file_path):
    vowels = 'aeiou'
    vowel_count = {v: 0 for v in vowels}

    try:
        with open(file_path, 'r') as file:
            content = file.read().lower()

            for char in content:
                if char in vowels:
                    vowel_count[char] += 1

        max_vowel = max(vowel_count, key=vowel_count.get)
        return max_vowel, vowel_count[max_vowel]

    except FileNotFoundError:
        return "File not found."

file_path = input("Enter the file path: ")
max_vowel, count = find_max_vowel_in_file(file_path)
print(f"The vowel with the maximum instances is '{max_vowel}' with {count} occurrences.")
```

```
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Mi
i Project)\Weeks\MCA-III_LAB> & C:/Users/Hammad/AppData/Local/Microsoft/WindowsApps/python3.12.e
e "c:/Users/Hammad/OneDrive - myamu.ac.in/Desktop/MCA/MCA III/CAMS3P01 Laboratory Course-III (Mi
i Project)/Weeks/MCA-III_LAB/Week-8/Q6.py"
Enter the file path: Week-8/vowels.txt
The vowel with the maximum instances is 'a' with 7 occurrences.
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Mi
i Project)\Weeks\MCA-III_LAB>
```

7. Write a Python program to create a list of user's supplied distinct integers having even number of elements. The program further creates two lists of equal lengths based on the original list, where first list is having all elements less than elements of second list. Display both the lists.

```
def create_even_split_lists():
    input_list = input("Enter distinct integers separated by spaces: ")
    integers = list(map(int, input_list.split()))

    if len(integers) % 2 != 0:
        print("The list must have an even number of elements.")
        return

    integers.sort()
    mid = len(integers) // 2

    L1 = integers[:mid]
    L2 = integers[mid:]

    less_than_list = [x for x in L1 if x < L2[0]]
    greater_than_list = [x for x in L2 if x > L1[-1]]

    print("List 1 (L1):", less_than_list)
    print("List 2 (L2):", greater_than_list)

create_even_split_lists()
```

```
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Mi
i Project)\Weeks\MCA-III_LAB> & C:/Users/Hammad/AppData/Local/Microsoft/WindowsApps/python3.12.e
e "c:/Users/Hammad/OneDrive - myamu.ac.in/Desktop/MCA/MCA III/CAMS3P01 Laboratory Course-III (Mi
i Project)/Weeks/MCA-III_LAB/Week-8/Q7.py"
Enter distinct integers separated by spaces: 1 3 5 2 6 4 8 7
List 1 (L1): [1, 2, 3, 4]
List 2 (L2): [5, 6, 7, 8]
PS C:\Users\Hammad\OneDrive - myamu.ac.in\Desktop\MCA\MCA III\CAMS3P01 Laboratory Course-III (Mi
```