Lab# 02
File and Directory Management

CS311L

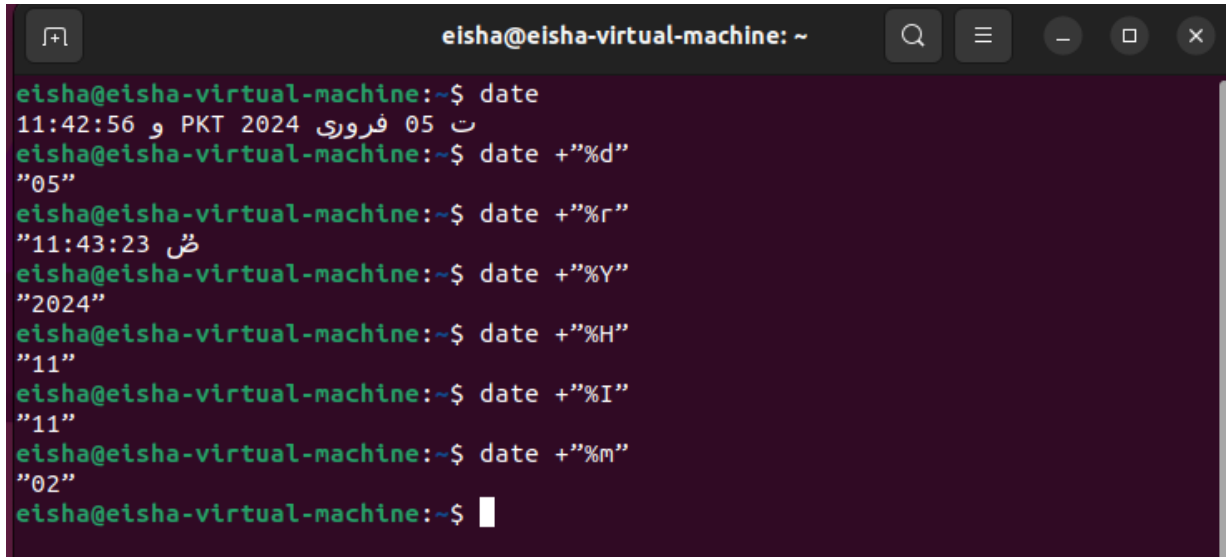
Engr. Eisha ter raazia Mir

Learning Objectives

- To set different permissions to a file and a directory (Read, Write and Execute).
- Set different permissions for different users (Owner, group, and others)
- List all the users on the system and display the user ID
- Use the manual page (**man**)
- Use wild cards

Linux Commands

- **date**
  - Displays the current date and time on the screen
- **date +”%d”**
  - Display just today’s date only
- **date +”%r”**
  - Display just time only
- **date +”%Y”**
  - Display just year only
- **date +”%H”**
  - Display just hours only
- **date +”%I”**
  - Display just hours only
- **date +”%m”**
  - Display just month only



Linux Commands

- **clear**
  - Clear the screen
- **echo**
  - Echoes back whatever you type on the command line after **echo**
- **echo -n anything**
  - **n** doesn't begin a new line after echoing the information

```
eisha@eisha-virtual-machine:~$ echo eisha
eisha
eisha@eisha-virtual-machine:~$ echo -n hello
helloeisha@eisha-virtual-machine:~$
```

Linux Commands

- **sort file_name**
  - Create the file ***sortos*** using ***nano*** and put the following content in that file
    - Hello this is eisha
    - file to be sorted
    - File To be Sorted
    - 22 years old
    - End of file
    - This is Os lab 2

Output ?

Linux Commands

- **sort - f file_name**
  - Ignores the distinction between lowercase and upper case letters
  - **sort - fr file_name/ sort -f -r file_name**
  - Reverse the order

sort --help

```
helloeisha@eisha-virtual-machine:~$ nano sortos
eisha@eisha-virtual-machine:~$ sort sortos
22 years old
End of file
file to be sorted
File To be Sorted
Hello this is eisha
This is Os lab 2
eisha@eisha-virtual-machine:~$ sort -f sortos
22 years old
End of file
file to be sorted
File To be Sorted
Hello this is eisha
This is Os lab 2
eisha@eisha-virtual-machine:~$ sort -fr sortos
This is Os lab 2
Hello this is eisha
File To be Sorted
file to be sorted
End of file
22 years old
eisha@eisha-virtual-machine:~$ sort -f -r sortos
This is Os lab 2
Hello this is eisha
File To be Sorted
file to be sorted
End of file
22 years old
eisha@eisha-virtual-machine:~$
```

Linux Commands

- **wc(word count) file_name**
  - Number of lines, number of words and number of characters in a file
- **Options**
- **-c** Display only the number of characters in the file
- **-l** Display only the number of lines in the file
- **-w** Display only the number of words in the file

```
eisha@eisha-virtual-machine:~$ wc sortos
 6 23 98 sortos
eisha@eisha-virtual-machine:~$ wc -w sortos
23 sortos
eisha@eisha-virtual-machine:~$ wc -c sortos
98 sortos
eisha@eisha-virtual-machine:~$ wc -l sortos
6 sortos
eisha@eisha-virtual-machine:~$
```

Linux Commands

- **who**
  - **who** command lists the login names, terminal lines, and login times of the users who are currently logged on to the system
- **whoami**
  - If you type **whoami**, Linux displays who the system thinks you are

```
eisha@eisha-virtual-machine:~$ who
eisha    tty2         2024-02-05 23:53 (tty2)
eisha@eisha-virtual-machine:~$ whoami
eisha
eisha@eisha-virtual-machine:~$
```

Linux Commands

- **head file_name**
  - This will print the first ten lines of the file if it contains more than ten lines
- **head –n 4 file_name**
  - This will print the first 4 lines of the file instead of first 10

head --help

```
eisha@eisha-virtual-machine:~$ head -n 4 sortos
Hello this is eisha
file to be sorted
File To be Sorted
22 years old
eisha@eisha-virtual-machine:~$
```

Linux Commands

- **tail file_name**
  - The **tail** command will, by default, write the last ten lines of the input file
- **tail –n4 file_name**
  - This will print the last 4 lines of the file instead of first 10

tail --help

File and Directory Security

Files/Directories can be created by setting permissions, allowing people to read, write, or execute your file.

- Each file on the machine divide – users – three categories:
  - The file's **owner** (who creates the file)
  - A **group** of users
  - **Other** users
  - **superuser**
- The system administrator - only superuser - several people have access to the superuser password
- Anyone logged in as superuser has access to every file directory

File Access permission Types

- There are three types of access:
  - **read**
  - **write**
  - **Execute**

Access Permission

- **Read** permissions:
  - Can be examined at a terminal, printed, viewed by an editor
- W**rite** permissions:
  - The file's contents can be changed (for example, by an editor)
  - File can be overwritten or delete

➢ Can change - access permission – your file

- ➢ Do not want anyone else to access a file - can remove read access for anyone but you
- ➢ Other users in your group to be able to write to a group of files, you can extend write permission to the group
- ➢ Each user type (the owner, the group, and others) can have any combination of the three access types for each file or Directory

Directory Access Permission Types

- Directory access permissions have different meanings:
  - **Read:** The read (**r**) permission in a directory means you can use the **ls** command to the filenames
  - **Write:** The write (**w**) permission in a directory means you can add or remove files from that Directory
  - **Execute:** The execute (**x**) permission in a directory means you can use the **cd** command to change to that Directory

Access Specification

- File or directory - default access specifications:
  - It may give all access permissions to the owner
  - Just read and write permissions to the group
  - Just read permission to everyone

drwx rw– r- -

user

group

others

- Nine places - divided - three sets - length 3
  - The first set of three - the **owner access**
    - The maximum access is represented by ***rwx***, indicating **read**, **write**, and **execute**
  - The next set of three - the **group access**
  - Final set of three - the access for everybody else
    - Whenever a dash (-) appears, access permission is not given

Checking Access

- A command can check the access privileges - files and directories:

ls -l file_name

?

Checking Permissions Commands

- id
  - It gives the user's name together with the groups they are a member, both names and numbers, and the user's user-id and current group-id.

Output ?

Change Permissions

- **chmod user-type [operations][permissions] filelist**
  - Change access permission for one or more files
  - **user-type**
    - **u** User or owner of file
    - **g** Group that owns the file
    - **o** Other
    - **a** All three user types
  - **operations**
    - **+** Add the permission
    - - Remove the permission
    - **=** Set permission; all other permission reset
  - **permissions**
    - **r** Read permission
    - **w** Write permission
    - **x** Execute(run) permission

Change Permissions

- *Change writing permission for user*

chmod u-w file_name

?

Write permission for owner (user) removed from file

Changing Permissions

chmod go+r file_name

Group and other users get read access for file stock.

chmod g=r + x file_name

It will set group access for reading and executing but not writing

?

Numeric Equivalent of Desired Permission

Owner			Group			Other		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1

Number	Permisison Type	Symbol
0	No Permission	---
1	Execute	--x
2	Write	-w-
3	Execute + Write	-wx
4	Read	r--
5	Read + Execute	r-x
6	Read + Write	rw-
7	Read + Write + Execute	rwx

Manual Page

- These manual pages provide detailed information about the command, including its options, usage, and examples.
- ***man command_name***
  - ***man command_name***

  man ls

Wildcards

- **Wildcards** are special characters used in Linux commands to match **multiple files or patterns** instead of typing names one by one.
- Operate on a group of files/directories
- Have some standard features in their names

Wildcards

- *****
  - It matches **zero or any** number of characters
  - For example, *.txt matches all files with a .txt extension (e.g., file.txt, document.txt).
- **?**
  - It matches **any single** character in a file/directory
  - For example, file?.txt matches file1.txt and fileA.txt, but not file12.txt.
- **[ ]**
  - It matches any one character in the bracket.
  - For example, file[1-3].txt matches file1.txt, file2.txt, and file3.txt.

Wildcard *

- **ls *t**
  - The last character should be 't' with any number of preceding characters

- **ls t***
  - The first character should be 't' with any number of following characters

- **ls t*t**
  - First and last characters should be 't' with any number of characters between them

- **ls f *t**
  - First and last characters should be 'f' and "t" respectively, with any number of characters between them

- **ls *oo**
  - The last two characters should be 'oo' with any number of preceding characters

Wildcard ?

- **ls t?**
  - The first character should be 't' with **only one** character following it
- **ls t?t**
  - First and last character should be 't' with **only one** character between them
- **ls ? t**
  - The last character should be 't' with **only one** character before it

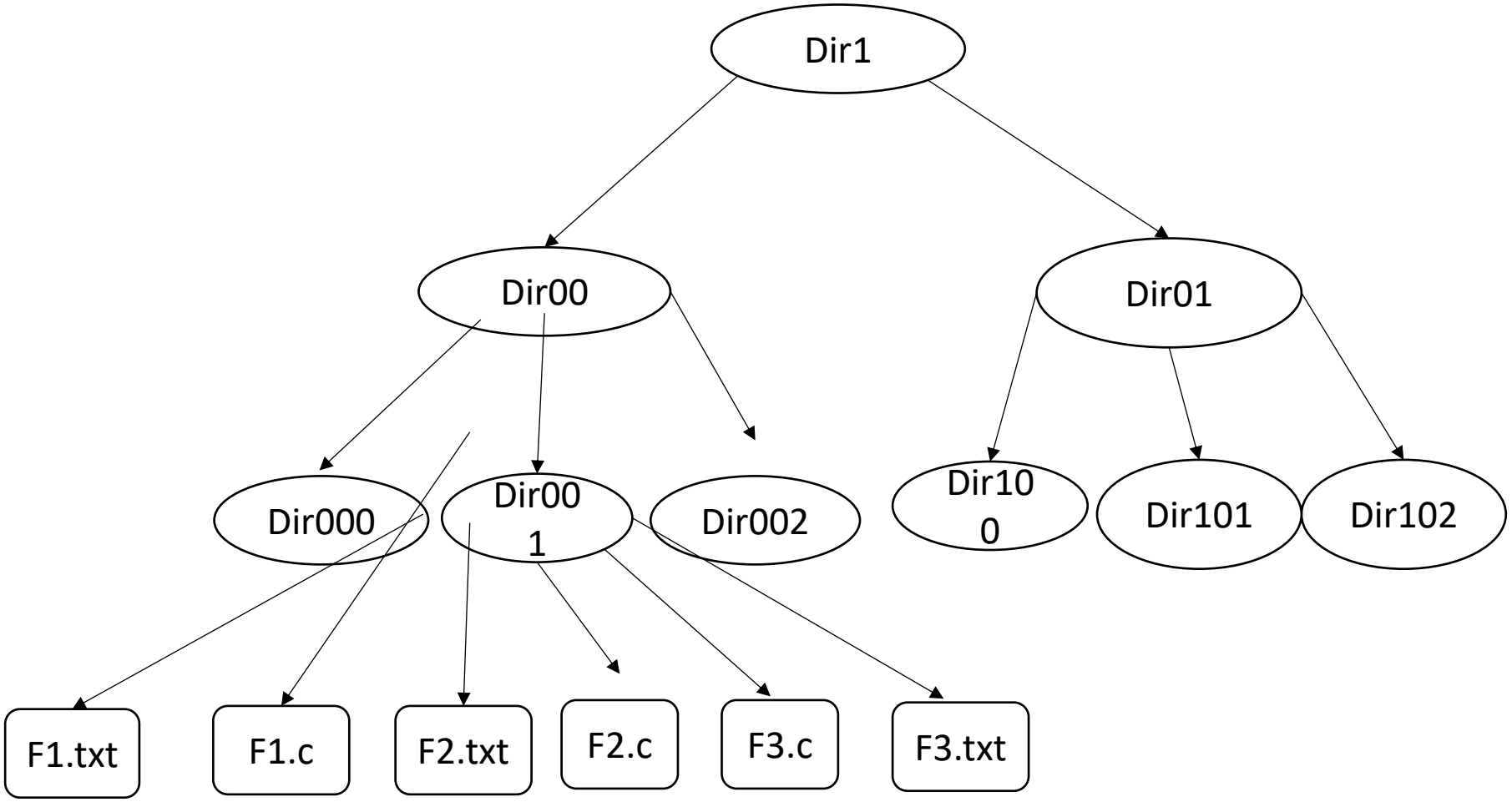
Wildcard []

- **ls p[12]**
  - Starting character should be 'p' and ending character should be '1' **or** '2'
- **ls p[1-9]**
  - Starting character should be 'p' and ending character could be anything between '1' **to** '9'
- **???t**
  - Four character file name. The first three characters may be any, but last character should be 't.'
- **??[a-c]**
  - Three character filename beginning with any two characters, but the last character should be 'a', 'b,' or 'c.'
- **[a-c][1-9]**
  - Two character file name starting with 'a', 'b', or 'c' and ending with any character between '1' and '9'

Wildcard

ls student*

?


Dir1
Dir00
Dir01
Dir000
Dir00
1
Dir002
Dir10
0
Dir101
Dir102
F1.txt
F1.c
F2.txt
F2.c
F3.c
F3.txt

Ready

- Create some files names:

test1, test2, process, eisha7, linux9, belief, first.txt, f1.txt

first.txt:

Ubuntu, ubuntu, uubuntu, Hello world

f1.txt:

subkhan, abroad123, see, September

Tasks

1. For file test1:
    a. Set ***read, write*** and ***execute*** permissions for ***group***
    b. Remove ***read, write*** permissions for ***owner***
2. For file test2:
    a. Allow ***write*** permissions for ***owner***, ***execute*** permission for ***group*** and ***read*** permission for ***others***
3. Find the files whose ***first five characters*** can be any but the last should be a number between (***5-9)***
4. Find the files whose ***first*** and ***last*** characters should be between ***(p – s)*** and anything between them

Tasks

5. Try to copy the files with extension .c from Dir001 to Dir101 by keeping yourself in Dr01
6. Try to remove .txt extension files from Dir001 keeping yourself in Dr00
7. Try to move f1.c from Dir001 to Dir1 by keeping yourself in Dir102

Task:

- **Create a file named studentfile.txt**
- **Modify the file's permissions**:

Change the permissions so that:

  - **Owner** can read, write, and execute the file.
  - **Group** can read and write the file.
  - **Others** can only read the file.
- **Remove all permissions for Others**
- **Grant execute permission to the Group**