

mkdir OS\_Course – Creates a directory named OS\_Course.

mkdir OS\_Lab – Creates a directory named OS\_Lab.

ls – Lists all files and directories in the current directory.

cd OS\_Lab – Changes the current directory to OS\_Lab.

mkdir lab\_Class\_Task – Creates a directory named lab\_Class\_Task inside OS\_Lab.

mkdir Lab\_Activities – Creates a directory named Lab\_Activities inside OS\_Lab.

mkdir lab\_Practice – Creates a directory named lab\_Practice inside OS\_Lab.

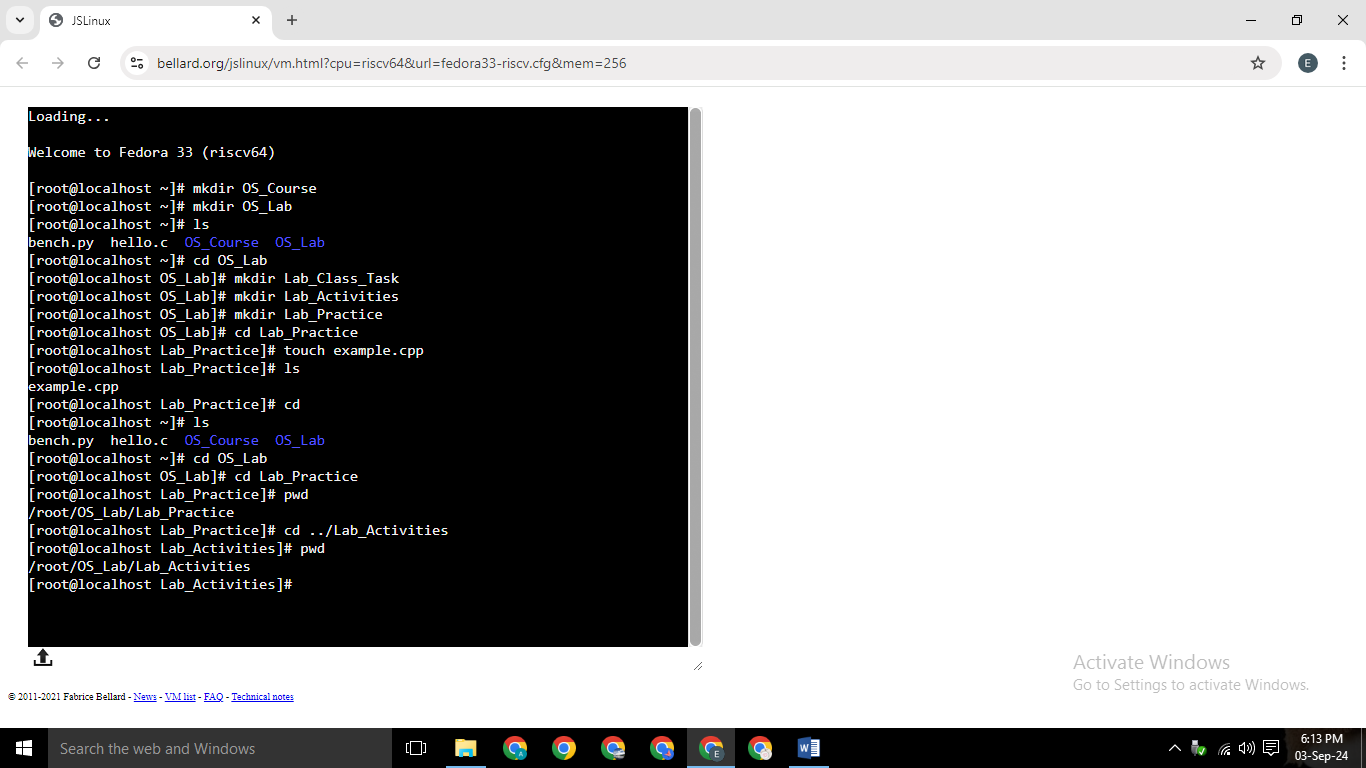
cd lab\_Practice – Changes the current directory to lab\_Practice.

touch example.cpp – Creates an empty file named example.cpp inside lab\_Practice.

ls – Lists all files and directories in the current directory (lab\_Practice).

cd – Returns to the parent directory or home directory (~).

ls – Lists all files and directories in the current directory again (the home directory).

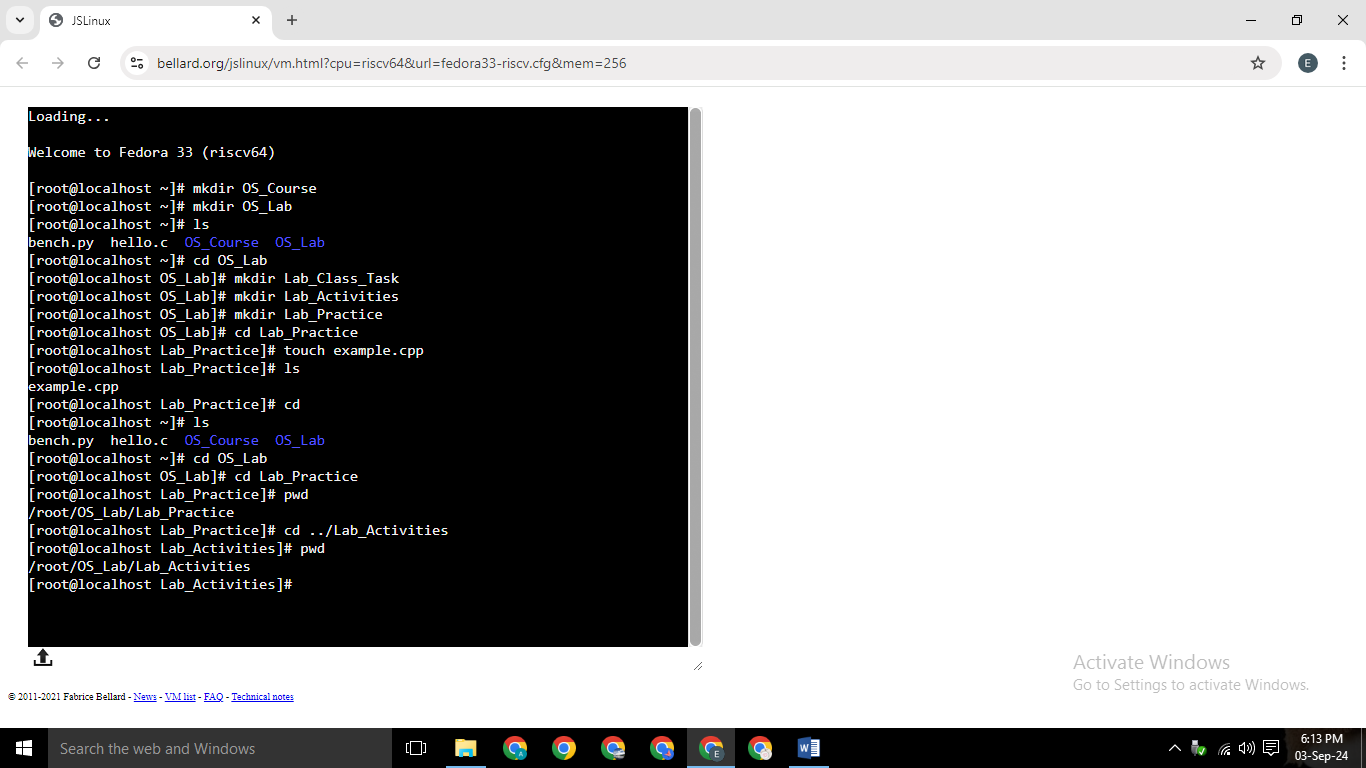


ls – Lists all files and directories in the current directory.

cd OS\_Lab – Changes the current directory to OS\_Lab.

cd Lab\_Practice – Changes the current directory to Lab\_Practice inside OS\_Lab.

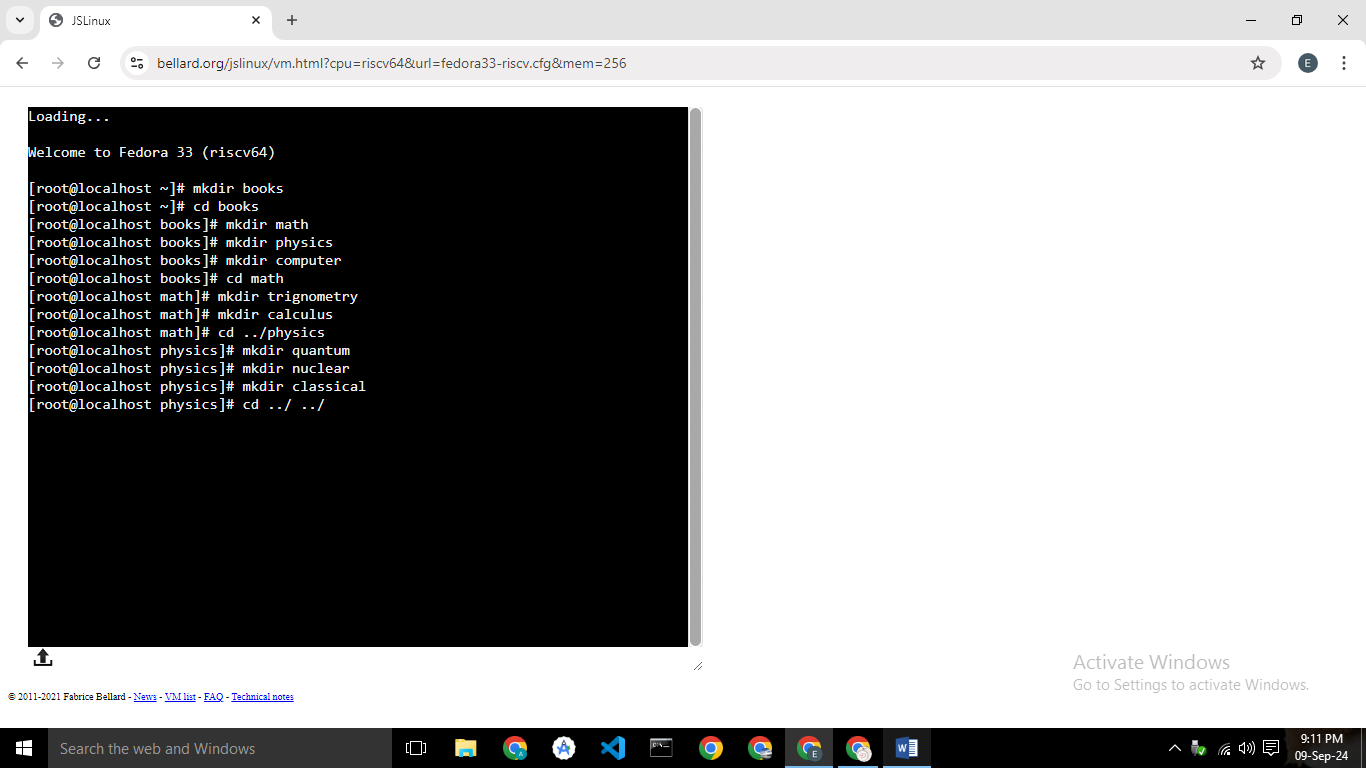
pwd – Prints the current working directory path, which is /root/OS\_Lab/Lab\_Practice.



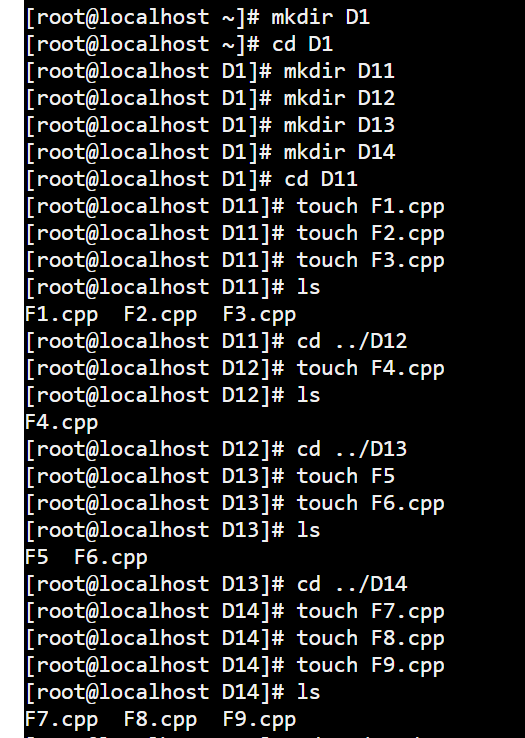
If the current directory is /home/user/:

documents/file.txt (relative path to /home/user/documents/file.txt)

../another\_folder/file.txt (relative path to /home/another\_folder/file.txt)

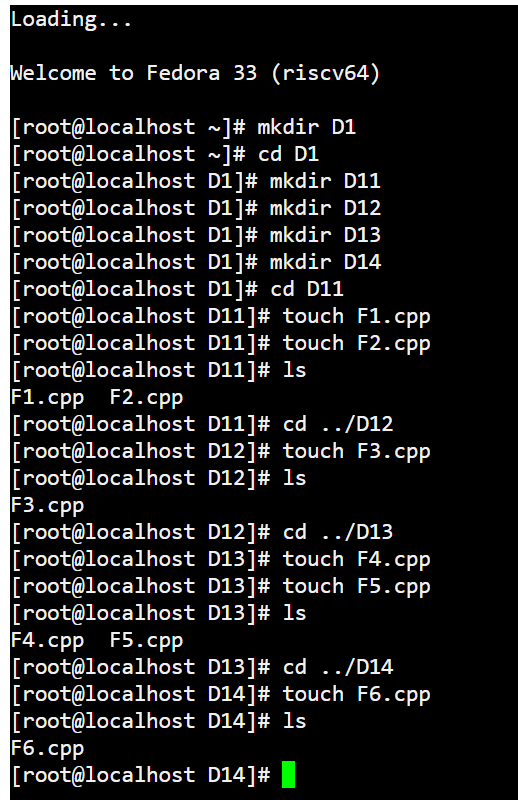


The commands create a directory structure for books, with subdirectories for math, physics, and computer, and further subdirectories within math and physics for specific topics.



The commands create a directory structure with the following hierarchy:

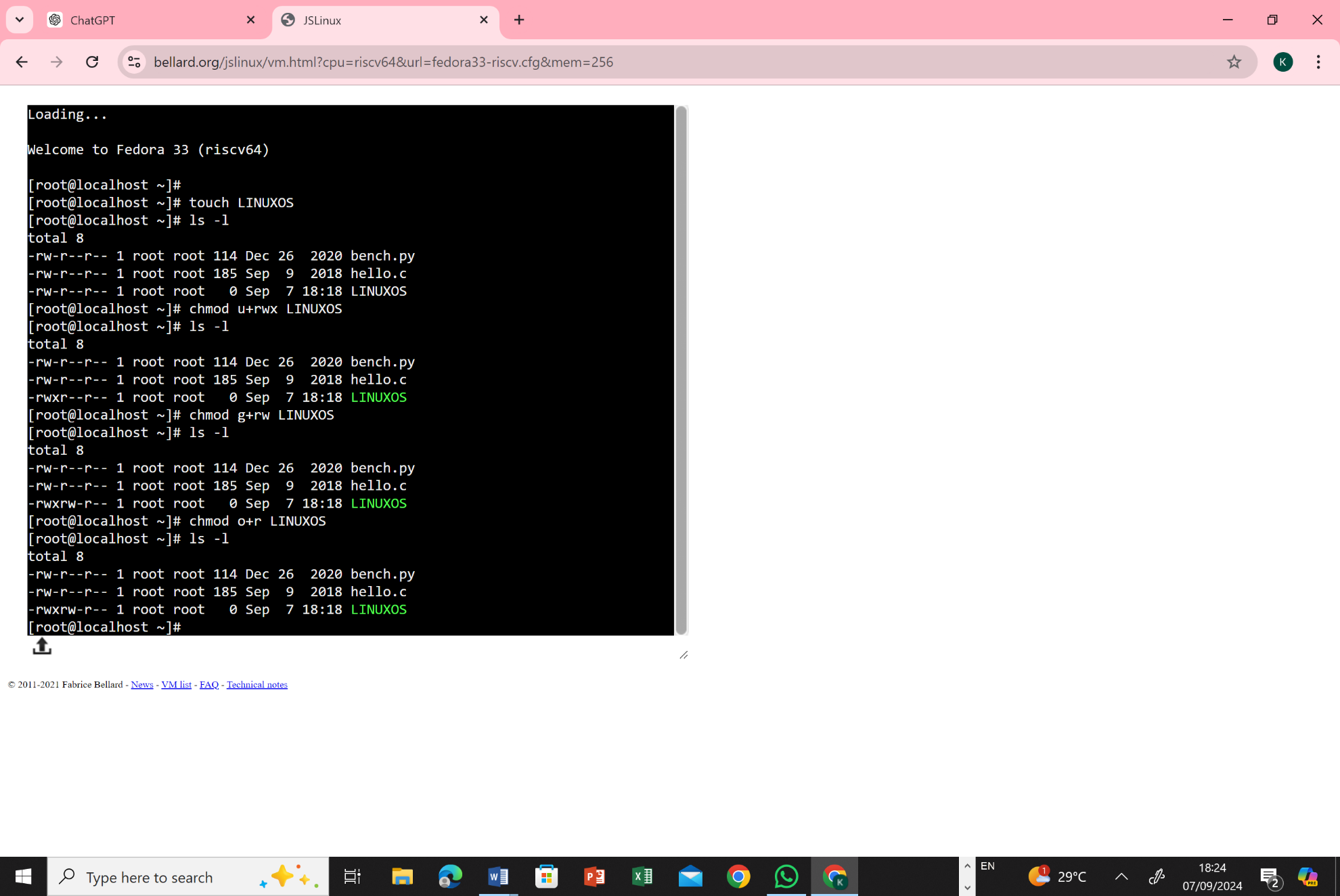
* **D1:** The root directory.
* **D11, D12, D13, D14:** Subdirectories of D1.
* **F1.cpp, F2.cpp, F3.cpp:** Files in D11.
* **F4.cpp:** File in D12.
* **F5, F6.cpp:** Files in D13.
* **F7.cpp, F8.cpp, F9.cpp:** Files in D14.
* **mkdir D1:** Creates a new directory named D1.
* **cd D1:** Changes the current directory to D1.
* **mkdir D11, D12, D13, D14:** Creates new subdirectories D11, D12, D13, and D14 within D1.
* **touch F1.cpp, F2.cpp, F3.cpp, F4.cpp, F5, F6.cpp, F7.cpp, F8.cpp, F9.cpp:** Creates empty files with the specified names in their respective directories.
* **ls:** Lists the contents of the current directory.
* **cd ..:** Changes the current directory to the parent directory.



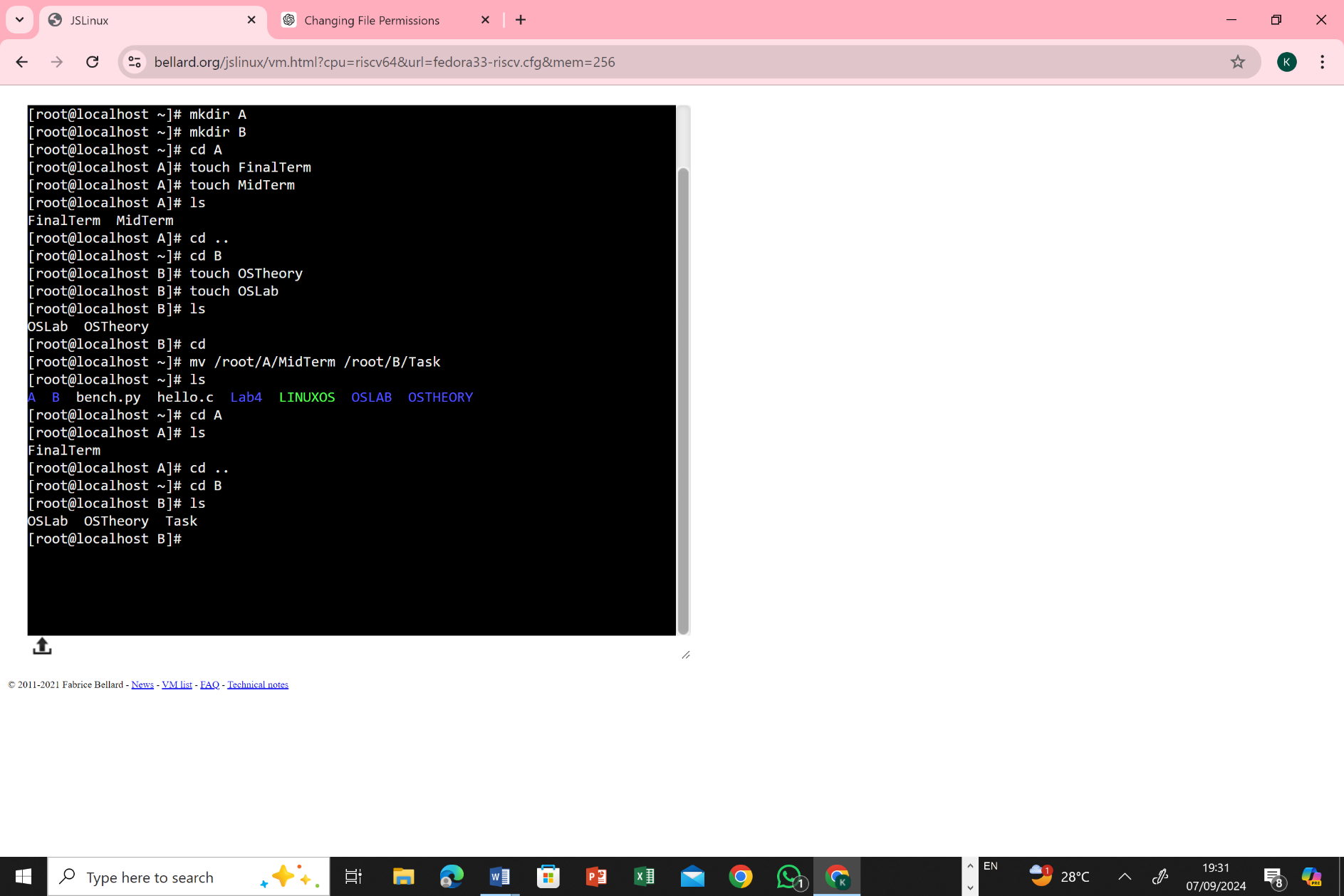
The commands create a directory structure with the following hierarchy:

* **D1:** The root directory.
* **D11, D12, D13, D14:** Subdirectories of D1.
* **F1.cpp, F2.cpp, F3.cpp:** Files in D11.
* **F4.cpp:** File in D12.
* **F5, F6.cpp:** Files in D13.
* **F6.cpp:** File in D14.
* **mkdir D1:** Creates a new directory named D1.
* **cd D1:** Changes the current directory to D1.
* **mkdir D11, D12, D13, D14:** Creates new subdirectories D11, D12, D13, and D14 within D1.
* **touch F1.cpp, F2.cpp, F3.cpp, F4.cpp, F5, F6.cpp:** Creates empty files with the specified names in their respective directories.
* **ls:** Lists the contents of the current directory.
* **cd ..:** Changes the current directory to the parent directory.

The final output of ls in the D14 directory shows that only the F6.cpp file exists, indicating that the rest of the files might have been deleted or renamed.



* -rw-r--r--: Permissions for the file:
* -: Regular file (not a directory or symbolic link)
* r: Read permission
* w: Write permission
* x: Execute permission
* -: No permission
* 1: Number of hard links to the file
* root: Owner of the file
* root: Group associated with the file
* 114: Size of the file in bytes
* Dec 26 2020: Modification date and time
* bench.py: Name of the file



The provided image shows a series of commands executed in a Linux terminal. Let's break down each command and its function:

**mkdir A:**

Creates a new directory named "A" in the current directory.

**mkdir B:**

Creates a new directory named "B" in the current directory.

**cd A:**

Changes the current directory to "A".

**touch FinalTerm:**

Creates a new file named "FinalTerm" in the current directory (which is now "A").

**touch MidTerm:**

Creates a new file named "MidTerm" in the current directory (which is still "A").

**ls:**

Lists the contents of the current directory, which are now the files "FinalTerm" and "MidTerm".

**cd ..:**

Changes the current directory to the parent directory of "A", which is the original directory.

**cd B:**

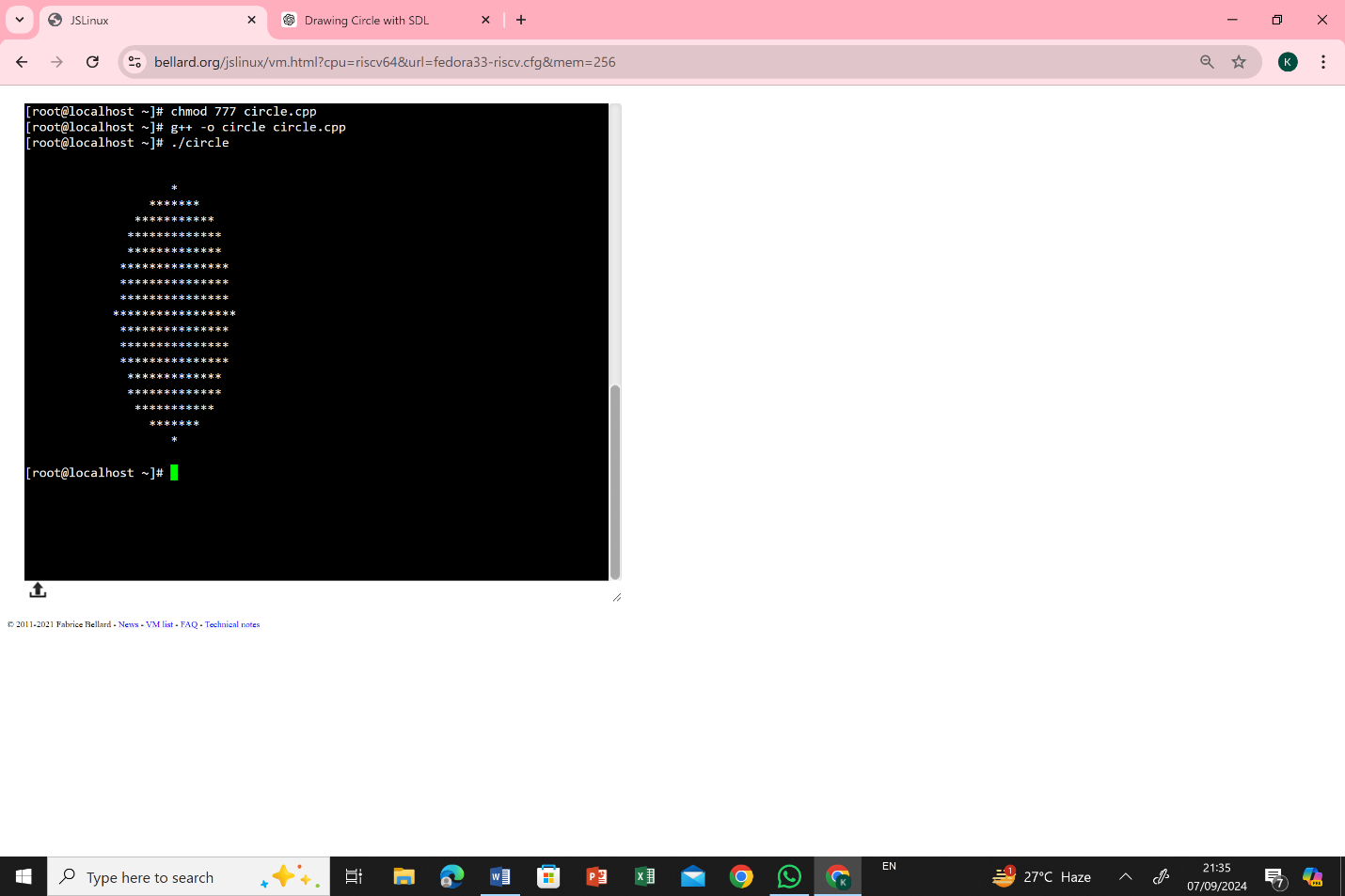
Changes the current directory to "B".

**touch OSTheory:**

Creates a new file named "OSTheory" in the current directory (which is now "B").

**touch OSLab:** - Creates a new file named "OSLab" in the current directory (which is still "B").

**ls:** - Lists the contents of the current directory, which are now the files "OSTheory" and "OSLab".



 chmod 777 circle.cpp:

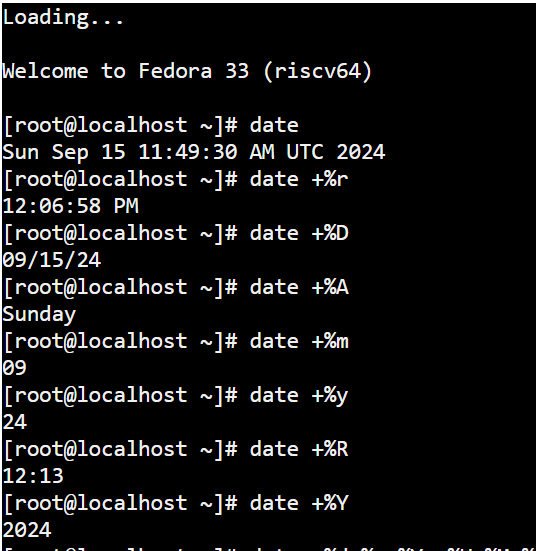
Modifies the permissions of the "circle.cpp" file to allow read, write, and execute permissions for the owner, group, and others. This is often done to ensure that the compiler can access and compile the file.

 g++ -o circle circle.cpp:

Compiles the C++ code in "circle.cpp" and creates an executable file named "circle". The -o option specifies the output filename.

 ./circle:

Executes the compiled C++ program named "circle".



1. date:

Displays the current date and time in UTC format.

1. date +%r:

Displays the current time in 12-hour AM/PM format with hours, minutes, and seconds.

1. 3. date +%D:

Displays the current date in the format month/day/year.

1. 4. date +%A:

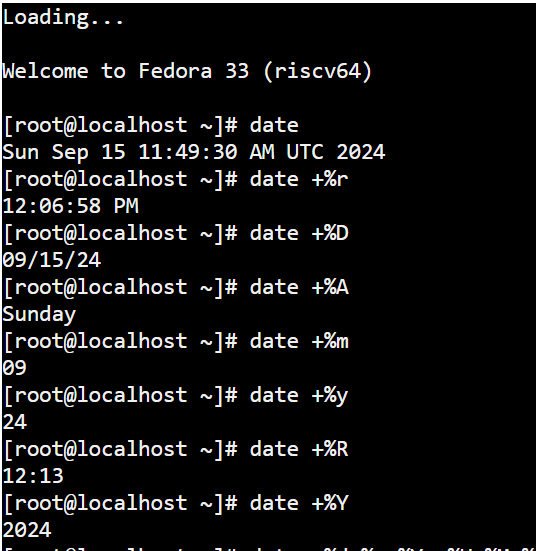
Displays the full name of the current day of the week.

1. 5. date +%m:

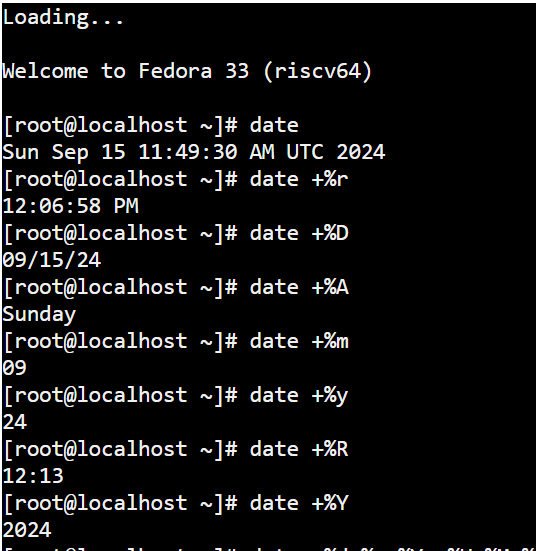
Displays the current month as a two-digit number (01 for January, 02 for February, etc.).

1. 6. date +%y:

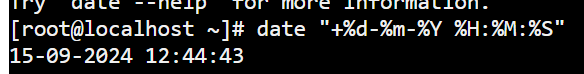
Displays the current year as a two-digit number (24 for 2024).



TO show time

****

To show current year

****

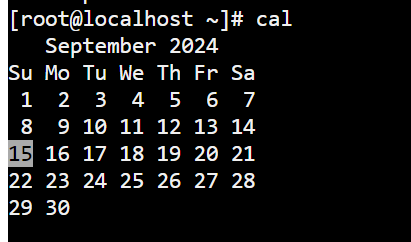
To show date month year and also show time in hours minutes and seconds



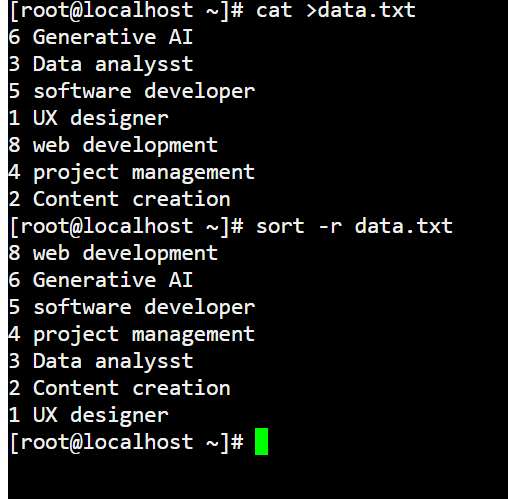
To show pervious date and time



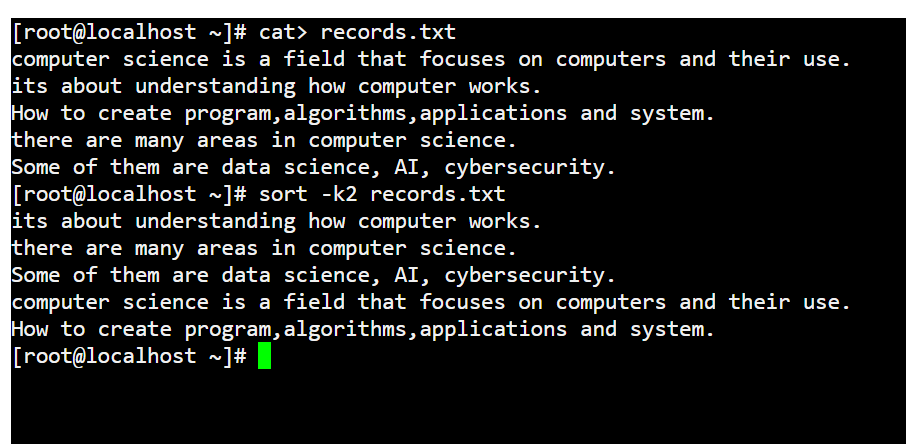
To show tomorrow time and date



To show the month calender



This command is use to reverse all text in descending order



* **cat > records.txt:** This command creates a new file named "records.txt" and allows you to enter text into it. You can type the text you want to store in the file and press Ctrl+D when you're finished.
* **cat records.txt:** This command displays the contents of the "records.txt" file.
* **sort -k2 records.txt:** This command sorts the lines of the "records.txt" file based on the second word in each line. It sorts the lines in ascending order.

The command used to compile a C program is **gcc** (GNU C Compiler). The basic syntax is:

**gcc eraj.c -o eraj**

"**eraj.c**" (only save files with .c extension only) be a file that contains the C code.

The **-o** option specifies the output file name for the compiled program. If not specified, the default output file name is **a.out**.

Example:

**gcc eraj.c -o eraj**

**chmod 755 myprogram (Give permission to all)**

Each file in Linux has permissions associated with three categories:

* **User (u)** – The owner of the file.
* **Group (g)** – The group to which the file belongs.
* **Others (o)** – Everyone else.

Permissions for each category are represented by three types:

* **Read (r)** – Allows viewing the file’s contents.
* **Write (w)** – Allows modifying the file.
* **Execute (x)** – Allows running the file as a program.