



Maestría en bioinformática
Principios y Lógica de Programación
Docente: Ing. María Gabriela Echeverría

Actividad 1: Creación del directorio de trabajo

Estudiante: Christian Robles

Fecha de entrega: 20 de Enero de 2026

1. Introducción

Esta actividad busca aplicar conceptos de organización en un *espacio de trabajo* dentro del sistema operativo **GNU/Linux**, como la creación de *directorios* y *archivos*, además de aprender a moverse dentro de ellos de manera práctica. La tarea la llevé a cabo en **Debian 13 Trixie**, utilizando el entorno de escritorio **XFCE**, **xfce4-terminal** y el gestor de archivos **Thunar**.

2. Creación del directorio de trabajo

Para comenzar el ejercicio, lo primero fue crear el directorio raíz del proyecto. Este directorio sigue la convención propuesta `User_project_year_publication`, en mi caso:

```
mkdir -v Robles_PhageOmics_2026_Science
cd Robles_PhageOmics_2026_Science
```

El comando `mkdir` sirve para crear directorios. La opción `-v` (verbose) muestra un mensaje del directorio que se creo, lo cual ayuda a verificar que se hizo correctamente. Luego usamos `cd` para entrar al nuevo directorio y empezar a organizar la estructura interna.

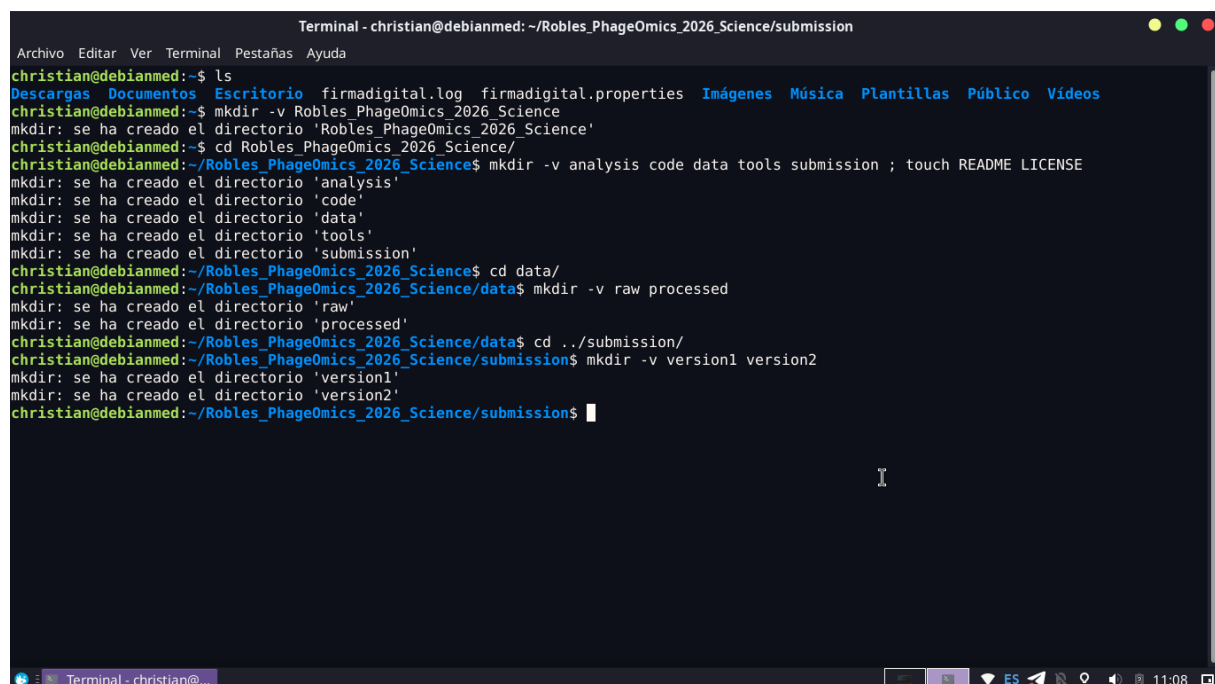


Figura 1: Creación del directorio principal del proyecto y de subdirectorios con el comando `mkdir`, creación de archivos iniciales con `touch`.

3. Creación de subdirectorios y archivos

Una vez dentro del directorio principal, organicé la estructura básica del proyecto. Como se muestra en la Figura 1.

Para ello, ejecuté los siguientes comandos:

```
mkdir -v analysis code data tools submission ; touch README LICENSE
```

El comando `mkdir` permite crear varios directorios a la vez, en este caso: `analysis`, `code`, `data`, `tools` y `submission`. El comando `touch` se usó para generar dos archivos vacíos: `README` y `LICENSE`.

4. Creación de subdirectorios adicionales

Dentro del directorio `data`, añadí dos carpetas `raw` y `processed` que serán útiles para organizar los datos del proyecto. Ver la Figura 1.

```
cd data
mkdir -v raw processed
```

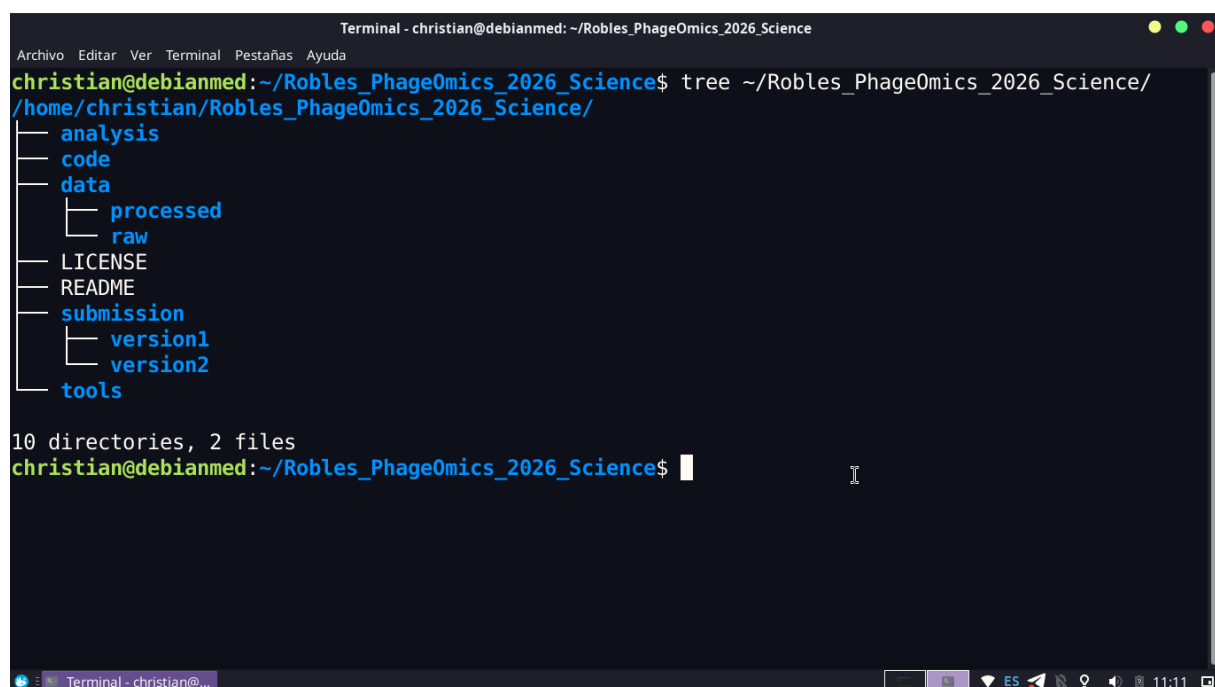
Luego, dentro del directorio `submission`, creé dos directorios `version1` y `version2`:

```
cd ../submission
mkdir -v version1 version2
```

5. Verificación de la estructura con `tree`

Después de crear todos los directorios y archivos, verifiqué que la estructura estuviera organizada correctamente. Para esto utilicé el comando:

```
tree ~/Robles_PhageOmics_2026_Science/
```



```

Terminal - christian@debianmed: ~/Robles_PhageOmics_2026_Science
Archivo Editar Ver Terminal Pestañas Ayuda
christian@debianmed:~/Robles_PhageOmics_2026_Science$ tree ~/Robles_PhageOmics_2026_Science/
/home/christian/Robles_PhageOmics_2026_Science/
├── analysis
├── code
├── data
│   ├── processed
│   └── raw
├── LICENSE
├── README
├── submission
│   ├── version1
│   └── version2
└── tools

10 directories, 2 files
christian@debianmed:~/Robles_PhageOmics_2026_Science$

```

Figura 2: Confirmación de la estructura de directorios con el comando `tree`

El comando `tree` muestra la estructura de carpetas y archivos en forma jerárquica. Esto permite confirmar de manera visual que los directorios se crearon de forma correcta de acuerdo al esquema planteado.

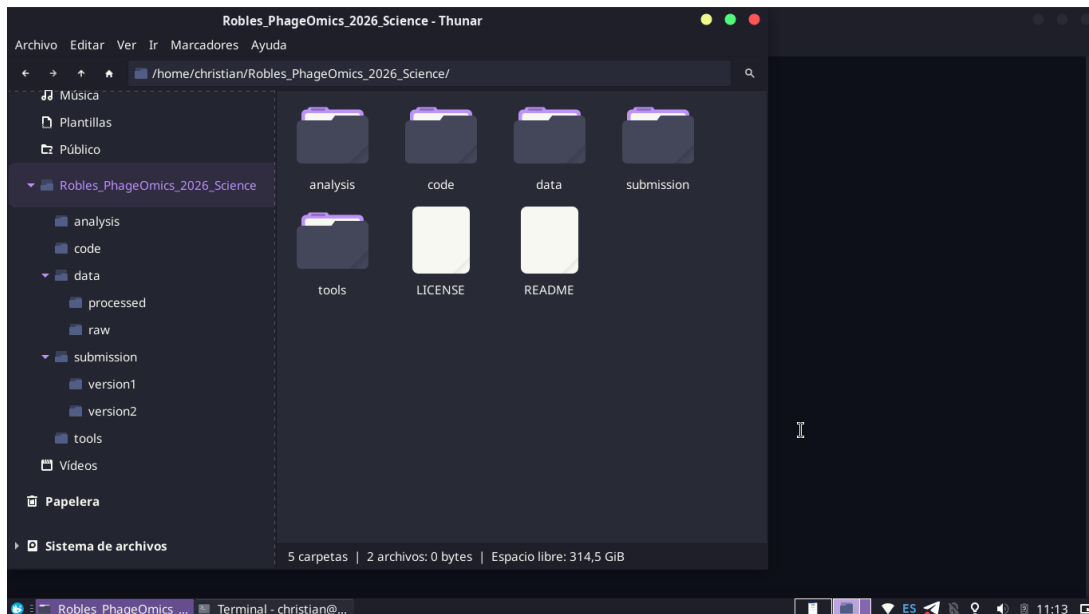


Figura 3: Estructura de directorios visualizados en el gestor de archivos Thunar.

6. Listado de inodos y permisos

Para revisar las características de cada archivo y directorio, utilicé el siguiente comando:

```
ls -lRhi
```

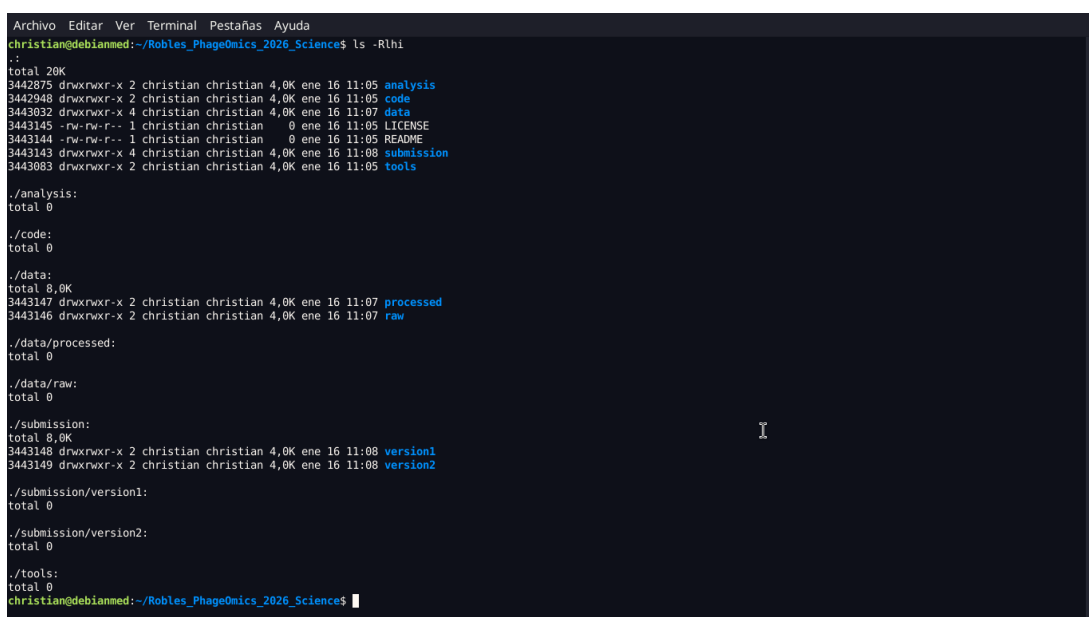


Figura 4: Inodos y permisos de los archivos del proyecto.

Este comando combina varias opciones:

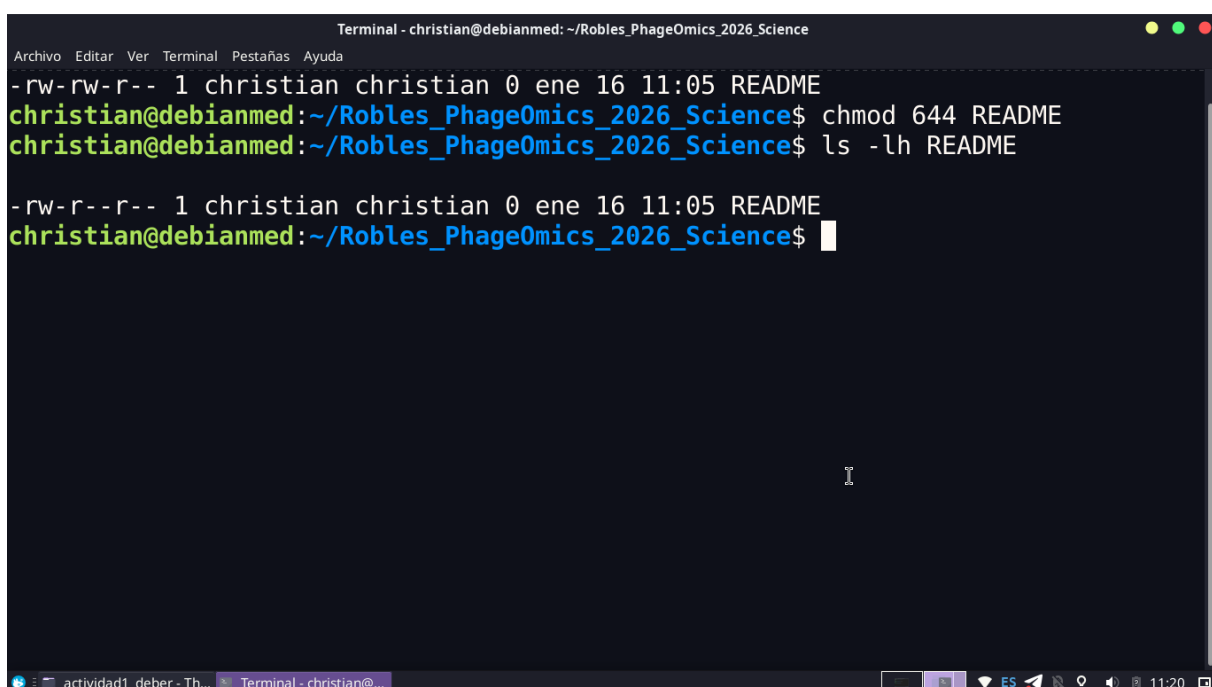
1. R → listado recursivo, muestra todo el contenido de subdirectorios.
2. l → formato detallado, incluye permisos, propietario y tamaño.
3. h → tamaños en formato legible (KB, MB).
4. i → muestra el número de inodo de cada archivo o carpeta.

De esta manera pude verificar tanto la estructura como los inodos y permisos de cada elemento del proyecto. Ver Figura 4

7. Cambio de permisos del archivo README

El archivo README debemos otorgarle permisos de lectura y escritura para el usuario, y solo lectura para grupo y otros. Para ello, ejecuté los comandos:

```
chmod 644 README
ls -lh README
```



```

Terminal - christian@debianmed: ~/Robles_PhageOmics_2026_Science
Archivo Editar Ver Terminal Pestañas Ayuda
-rw-rw-r-- 1 christian christian 0 ene 16 11:05 README
christian@debianmed:~/Robles_PhageOmics_2026_Science$ chmod 644 README
christian@debianmed:~/Robles_PhageOmics_2026_Science$ ls -lh README

-rw-r--r-- 1 christian christian 0 ene 16 11:05 README
christian@debianmed:~/Robles_PhageOmics_2026_Science$

```

Figura 5: Cambio de permisos del archivo README con chmod

El comando `chmod 644` asigna los permisos de la siguiente forma:

- 6 → usuario: lectura y escritura.
- 4 → grupo: solo lectura.
- 4 → otros: solo lectura.

Finalmente, con `ls -lh README` confirmé que los permisos se aplicaron correctamente, mostrando `-rw-r--r--`.

8. Conclusiones

En esta actividad organizamos un directorio de trabajo para el proyecto, creando subcarpetas que nos ayudan a mantener todo en orden y encontrar rápido nuestros archivos.

Se usaron comandos básicos como `mkdir`, `touch`, `tree`, `ls` y `chmod` para construir la estructura de nuestro espacio de trabajo, revisarla y ajustar permisos. Esta organización facilita que cualquier persona entienda el proyecto y refuerza la importancia de mantener el orden desde el inicio.

9. Bibliografía

- Fox, B., & Ramey, C. (2025). *GNU Bash Reference Manual* [Consultado en enero de 2026]. <https://www.gnu.org/software/bash/manual/>
- Newham, C., & Robbins, A. (2005). *Learning the Bash Shell* (3rd). O'Reilly Media.
- Shotts, W. E. (2019). *The Linux Command Line: A Complete Introduction* (2nd). No Starch Press.