



# Universidad Nacional Autónoma de México

---

**Proyecto 02 - (Micro) sistema de archivos  
multihilos**

**Curso: Sistemas Operativos**

**Profesor: Dr. Gunnar Eyal Wolf Iszaveich**

**Autor: Chávez López B. Alejandro**

**Cuenta: 421112601**

**Grupo: 08**

November 21, 2025

# 1 Descripción del Proyecto

El objetivo de esta práctica fue crear un programa capaz de manipular el sistema de archivos *FiUnamFS*. La idea principal es simular el funcionamiento de un disquete de 1.44 MB, interactuando directamente con los bytes del archivo `fiunamfs.img` sin depender del sistema operativo.

El proyecto consistió en implementar funciones para listar, extraer y eliminar archivos. Además, para la escritura de nuevos archivos, se debía cumplir con el requisito de usar programación concurrente (hilos) para transferir la información.

## 2 Herramientas Utilizadas

Para el desarrollo utilicé el lenguaje **Python**. La elección se debió a que es un lenguaje con el que ya he desarrollado proyectos anteriores y me resulta familiar. Además, cuenta con librerías nativas como `struct` (para manejar datos binarios) y `threading` (para los hilos) que facilitaron la implementación sin complicar el código innecesariamente.

## 3 Detalles de la Implementación

### 3.1 Lectura de Datos

Dado que el sistema utiliza el formato *Little Endian*, fue necesario utilizar funciones de empaquetado y desempaquetado de bytes constantemente. Me basé en las especificaciones del documento para definir el tamaño del cluster en 1024 bytes y las entradas de directorio en 64 bytes.

### 3.2 Escritura y Espacio Libre

Para guardar archivos, implementé una lógica de asignación contigua. El programa revisa todo el directorio para mapear qué clusters están ocupados y busca el primer "hueco" continuo donde quepa el archivo que se quiere copiar. Si no hay espacio continuo suficiente, la operación se cancela.

### 3.3 Uso de Hilos

Para cumplir con la parte de concurrencia, utilicé el modelo de Productor-Consumidor. Un hilo se encarga de leer el archivo de mi computadora y poner los datos en una cola, mientras que el segundo hilo saca esos datos y los escribe en la imagen del disco. Esto permite que la lectura y escritura ocurran de forma sincronizada.

### 3.4 Pantallazos de pruebas

```
Iniciando sistema
SISTEMA DE ARCHIVOS DETECTADO
Nombre: FiUnamFS
Versión: 26-1
Etiqueta: Mi nombre.
Tamanioo de Cluster: 1024 bytes
Clusters de Directorio: 3
Clusters Totales: 1440

[Correcto] El sistema de archivos es VALIDO

=====
SISTEMA DE ARCHIVOS FIUNAMFS
=====
1. Listar contenidos del directorio
2. Copiar archivo DE FiUnamFS A tu PC
3. Copiar archivo DE tu PC A FiUnamFS
4. Eliminar archivo del FiUnamFS
5. Salir
-----
Selecciona una opcion: █
```

Figure 1: Pantalla principal

## 4 Problemas Encontrados y Soluciones

Durante el desarrollo surgieron varios detalles técnicos que requirieron ajustes en el código:

- **Versión del Sistema:** Al principio, el validador fallaba porque la documentación indicaba la versión "26-2", pero el archivo .img descargado tenía la versión "26-1". Tuve que ajustar la validación para aceptar ambas.
- **Limpieza de Nombres:** Tuve problemas al comparar los nombres de los archivos porque el método `strip()` no eliminaba los caracteres nulos (0x00) que el sistema usa como relleno. Esto se solucionó limpiando primero los nulos y después los espacios.

- **Archivos Duplicados:** Noté que el programa permitía guardar el mismo archivo varias veces, creando entradas repetidas. Para corregirlo, agregué una verificación al inicio de la función de copiado que revisa si el nombre ya existe antes de empezar a escribir.

DESPUES DE ELIMINAR		
NOMBRE	TAMAÑO	CLUSTER INICIAL
<hr/>		
README.org	31104	5
logo.png	120857	36
prueba2.txt	44	250
prueba3.txt	44	251
saludo.jpg	95307	155
prueba4.txt	44	252
prueba.txt	44	253
prueba.txt	44	254

Figure 2: Captura de pantalla mostrando duplicados

- **Longitud del Nombre:** Implementé una restricción de 15 caracteres para los nombres. Si se permitían nombres más largos, se sobrescribía el byte donde inicia el cluster, corrompiendo el archivo.

## 5 Conclusiones

Realizar este proyecto me permitió comprender mejor cómo se estructuran los datos dentro de un sistema de archivos a bajo nivel. Aunque la lógica general de copiar y borrar parece sencilla, manipular directamente los bytes requiere mucho cuidado con los offsets y los tipos de datos.

La implementación de los hilos funcionó correctamente para la transferencia de datos. Como mejora, el sistema valida correctamente errores comunes (como falta de espacio o nombres duplicados), resultando en una herramienta funcional para manipular imágenes *FiUnamFS*.