

filas, columnas : Dimensiones de la imagen

IMG\_original: Almacena la imagen original (Triple puntero para RGB)

IMG\_transformada: Almacena la imagen durante las transformaciones.

IMG\_aleatoria: Imagen aleatoria para operaciones XOR.

mascara: La mascara a usar para el enmascaramiento de la imagen.

Operaciones Basicas: cargar\_IMG, guardar\_IMG

Transformaciones: aplicarXOR. rotarBITS, aplicar\_mascara

Logica Principal: revertir\_TRANSFOR

Verificación: verificar\_mascara

Este, sería el programa “Ideal” según los requisitos planteados en el PDF.

A el programa planteado, le falta lo siguiente:

**Operaciones de bits:**

Función para aplicar XOR entre imágenes (Io e IM)

Función para rotar bits (hasta 8 bits)

Función para desplazamientos de bits

**Procesamiento de transformaciones:**

Algoritmo para revertir las transformaciones en orden inverso

Implementación del enmascaramiento (suma con la máscara M)

Función para verificar los archivos de enmascaramiento (M1.txt, M2.txt, etc.)

**Lógica principal:**

Mecanismo para determinar el orden correcto de las transformaciones aplicadas

Implementación del proceso de reconstrucción de la imagen original

**Manejo de datos:**

Carga y procesamiento de múltiples archivos de enmascaramiento (no solo M1.txt)

Manejo de la imagen aleatoria (IM) y la máscara (M)

**Validaciones:**

Verificación de consistencia entre operaciones y archivos de rastreo

Mecanismo para probar diferentes combinaciones de operaciones

**Optimización:**

Liberación adecuada de memoria para todas las estructuras dinámicas

Manejo de errores para casos donde las transformaciones no coincidan

## 21/04/2025

Decidí, debido a que no entendí el codigo hecho por el profesor, reescribir el codigo desde cero, basandome en los requisitos.

## Funciones a crear:

unsigned char\* cargar\_imagen(const char\* ruta, int& ancho, int& alto);

bool guardar\_imagen(unsigned char\* datos, int ancho, int alto, const char\* ruta);

void aplicar\_xor(unsigned char\* imagen1, unsigned char\* imagen2, int tamaño);

void rotar\_bits(unsigned char\* imagen, int tamaño, int bits);

unsigned int\*\* cargar\_archivos\_enmascaramiento(const char\*\* rutas, int num\_archivos, int\* semillas, int\* num\_pixeles);

void liberar\_datos(unsigned int\*\* datos, int num\_archivos);

void procesar\_imagen(unsigned char\* io, unsigned char\* im, unsigned char\* m, int ancho, int alto);