**PARCIAL 1**

**Encriptación XOR**

**Integrantes:** *Vladlen Shatunov*

**1.Justificación de la Implementación de lectura de archivo y reconocimiento del tipo de archivo**

Se utiliza la verificación de cada byte en el archivo para determinar si es de texto o binario  
Se considera que un archivo es de texto si todos sus bytes corresponden a caracteres imprimibles o a controles comunes como tabulador, salto de línea y retorno de carro  
Esta aproximación es suficiente para la mayoría de archivos de texto en formato ASCII  
El procesamiento se realiza a nivel de bytes lo que permite que la operación XOR se aplique de forma uniforme a cualquier tipo de archivo  
además, la detección del tipo de archivo informa al usuario y ayuda en la interpretación del contenido antes de proceder a la encriptación  
Esta decisión garantiza flexibilidad y compatibilidad con archivos de diferentes formatos sin necesidad de librerías externas

**2.Estrategia de almacenamiento y recuperación del archivo encriptado**.

La estrategia de almacenamiento y recuperación se basa en leer el archivo en modo binario y almacenarlo en un vector de bytes. Con este enfoque se pueden aplicar operaciones a nivel de byte sin depender de librerías externas.

Durante el proceso de encriptación se recorre el vector y se aplica la operación XOR a cada byte usando una clave fija. Una vez encriptados los datos se escribe el contenido resultante en un nuevo archivo usando la llamada al sistema write. De esta forma se genera un archivo encriptado que contiene la información transformada.

Para la recuperación se realiza el proceso inverso: se lee el archivo encriptado en modo binario y se almacena su contenido en el vector. Luego se vuelve a aplicar la misma operación XOR sobre cada byte; debido a la propiedad simétrica del XOR, esto devuelve el contenido original. Finalmente se escribe el resultado en un archivo de salida.

**3.Instrucciones de uso**

**Compilación**

asegurándonos de estar en el directorio correcto ejecutamos el siguiente comando para compilar nuestros archivos (main.cpp, ManejadorArchivo.h, ManejadorArchivo.cpp y Makefile):

make

Si la compilación es exitosa, se generará un archivo ejecutable llamado UsoBits.

**Ejecución del programa**

./UsoBits <bandera> <archivo\_entrada> [archivo\_salida]

* Donde <bandera> puede ser:
* **-h** o **--help**: Muestra la ayuda con las banderas disponibles.
* **-v** o **--version**: Muestra la versión del programa.
* **-e** o **--encrypt**: Encripta el archivo de entrada.
* **-d** o **--decrypt**: Desencripta el archivo de entrada.
* <archivo\_entrada> es el archivo a procesar (texto o binario).
* [archivo\_salida] es opcional; si no se especifica, se usará salida.bin como nombre por defecto.

**Ejemplo de uso**

* **Mostrar la ayuda**:

./UsoBits -h

*Muestra un listado de todas las banderas y su uso.*

* **Mostrar la versión**:

./UsoBits -v

*Indica la versión actual del programa.*

* **Encriptar un archivo** (por ejemplo, archivo.txt o imagen.png):

./UsoBits -e archivo.txt archivo\_encriptado.bin

O

./UsoBits -e imagen.png imagen\_encriptada.bin

*Esto leerá archivo.txt, aplicará la encriptación XOR y guardará el resultado en archivo\_encriptado.bin. El programa también indicara si el archivo se considera de texto o binario.*

* **Desencriptar un archivo** (por ejemplo, archivo\_encriptado.bin o imagen\_encriptada.bin):
* ./UsoBits -d archivo\_encriptado.bin archivo\_desencriptado.txt

O

./UsoBits -d imagen\_encriptada.bin imagen\_desencriptada.png

*Esto aplicara la misma operación XOR para recuperar el contenido original, que se guardara en archivo\_desencriptado.txt.*

**Verificación**

Para confirmar que el contenido desencriptado coincide con el original, utilizamos herramientas como md5sum o diff. Por ejemplo:

md5sum archivo.txt archivo\_desencriptado.txt

diff archivo.txt archivo\_desencriptado.txt

o

md5sum imagen.png imagen\_desencriptada.png

diff imagen.png imagen\_desencriptada.png

También puedes verificar como el archivo encriptado ya no es el mismo que el original aplicando el comando

md5sum archivo.txt archivo\_encriptado.bin

O

md5sum imagen.png imagen\_encriptada.bin

**Consideraciones**

* El programa detecta si el archivo se considera texto o binario revisando los bytes. Esto no afecta la encriptación (que se aplica de la misma manera) pero sirve como información adicional para el usuario.
* La encriptación se realiza a nivel de bytes, por lo que es posible encriptar cualquier tipo de archivo (texto, imagenes, videos, etc.).