**Informe de Desarrollo del Proyecto**

El proyecto consiste en hacer ingeniería inversa sobre un archivo que está encriptado y comprimido, con el fin de recuperar el mensaje original. El reto de este proyecto está en la administración correcta del heap para hacer un uso eficiente de los recursos computacionales y garantizar que el programa sea óptimo.

La información fue comprimida y encriptada con los siguientes criterios:

* Compresión mediante el método LZ78 o el método RLE
* Encriptación del mensaje aplicando dos operaciones consecutivas sobre cada byte:
  + Rotación de bits hacia la izquierda n posiciones.
  + Operación XOR con una clave K de un byte.

El resultado de estas transformaciones es el archivo único suministrado por el cliente, junto con un fragmento del texto original que sirve como pista.

**Estrategia de solución:**

Para abordar este problema, se plantea la siguiente metodología:

* Lectura controlada del archivo: el archivo encriptado se leerá en modo binario y se almacenará en un arreglo dinámico de tipo unsigned char\*, a fin de garantizar el manejo correcto de valores entre 0 y 255. La pista se almacenará en un arreglo dinámico de tipo char\*, por corresponder a texto plano ASCII.
* Desencriptación por fuerza bruta: se probarán todas las combinaciones posibles de parámetros (rotaciones de 1 a 7 bits y claves XOR de 0 a 255). Para cada combinación se aplicarán las operaciones inversas: XOR con la clave y rotación de bits hacia la derecha.
* Descompresión por doble vía: una vez desencriptado el contenido, se intentará descomprimir con RLE y con LZ78. Solo uno de los algoritmos producirá un texto válido.
* Comparación con la pista: el resultado descomprimido se confrontará con la pista conocida. Si aparece de forma exacta, se concluye que los parámetros de desencriptación y el método de compresión utilizados son correctos.
* Reconstrucción final: con los parámetros identificados, se procede a recuperar la totalidad del mensaje original.

La elección de esta metodología responde a que los parámetros de encriptación no son infinitos y las alternativas de compresión son solo dos. Por lo tanto, es viable aplicar una búsqueda por fuerza bruta sin que el número de combinaciones haga inviable la solución.

**División de tareas**

1. Lectura de los archivos de entrada

* Leer el archivo encriptado en modo binario y almacenarlo en un arreglo dinámico de tipo unsigned char\*.
* Leer el archivo de la pista en texto plano y almacenarlo en un arreglo dinámico de tipo char\*.

1. Desencriptación del archivo

* Aplicar de manera sistemática las operaciones inversas:
  + Operación XOR con una clave K candidata.
  + Rotación de bits hacia la derecha en n posiciones.
* Probar todas las combinaciones posibles de parámetros (1 ≤ n ≤ 7, 0 ≤ K ≤ 255).
* Guardar el resultado desencriptado en un arreglo temporal dinámico para posteriores pruebas de descompresión.

1. Descompresión del resultado desencriptado

Es muy importante verificar cual método corresponde, primero aplicar un metodo y comparar:

* Implementar un algoritmo de descompresión RLE, que reconstruya el mensaje a partir de pares (longitud, símbolo).

Si ese método no coincide con el texto original, aplicar si el otro.

* Implementar un algoritmo de descompresión LZ78, que reconstruya el mensaje a partir de pares (índice, carácter) y un diccionario dinámico.

1. Comparación con la pista

* Implementar una función que verifique si la pista se encuentra dentro del texto descomprimido.
* En caso de coincidencia exacta, concluir que los parámetros de desencriptación y el algoritmo de compresión han sido identificados correctamente.

**Algo que seria favorable seria poder aplicar esta función con la anterior.**

1. Reconstrucción del mensaje original

* Utilizar la combinación validada para desencriptar y descomprimir el archivo completo.
* Almacenar el resultado en un arreglo dinámico de caracteres.
* Generar como salida el mensaje recuperado y los parámetros identificados (método de compresión, rotación de bits y clave XOR).

Algoritmos que debemos realizar.

1. Algoritmo de rotación de bits (inversa)

* Entrada: un byte (unsigned char) y un valor n.
* Proceso: rotar el byte hacia la derecha n posiciones, reinsertando los bits desplazados en la parte izquierda.
* Salida: el byte original antes de la encriptación.

1. Algoritmo de desencriptación XOR (inverso)

* Entrada: un byte (unsigned char) y una clave K (0 ≤ K ≤ 255).
* Proceso: aplicar XOR entre el byte y la clave para recuperar el valor original.
* Salida: el byte desencriptado.

1. Algoritmo de descompresión RLE

* Entrada: pares (longitud, símbolo).
* Proceso: repetir el símbolo tantas veces como indique la longitud.
* Salida: texto descomprimido en un arreglo dinámico de caracteres.

1. Algoritmo de descompresión LZ78

* Entrada: pares (índice, carácter).

1. Algoritmo de comparación con la pista

* Entrada: texto descomprimido y arreglo de la pista.
* Proceso: buscar la aparición exacta de la pista dentro del texto.
* Salida: un valor booleano que indica si la pista fue encontrada.