

# Documentación Técnica – Proyecto UnCompressor

## 1. Descripción general del sistema

El proyecto UnCompressor es un sistema capaz de comprimir y descomprimir archivos .txt utilizando códigos de Huffman. Está estructurado a través de librerías independientes que encapsulan las funcionalidades principales del sistema: manejo de diccionarios de códigos, generación y uso del árbol de Huffman, y lectura estructurada de archivos binarios.

## 2. Librerías del proyecto

El sistema está diseñado alrededor de tres librerías principales:

### 2.1. Librería Dictionary

Rol: Gestiona las relaciones entre códigos Huffman y caracteres.

Estructuras abstractas utilizadas:

- Diccionarios hash: Acceso rápido para mapear códigos variables a caracteres.
- Conjuntos de prefijos: Permiten verificar fragmentos válidos sin un trie explícito.
- Árbol conceptual: Representa la estructura lógica del código sin almacenarla físicamente.

Funciones principales:

- insert()
- isPrefix()
- exists()

### 2.2. Librería Huffman

Rol: Implementa el algoritmo de Huffman para compresión y descompresión.

Estructuras abstractas utilizadas:

- Árbol binario: Representa combinaciones de símbolos mediante nodos.
- Cola de prioridad: Selección óptima de elementos de menor frecuencia.
- Vectores de códigos: Almacenan secuencias binarias generadas.

Funciones principales:

- buildTree()

- generateCodes()
- compress()
- decompress()

## 2.3. Librería Lector

Rol: Gestiona la lectura binaria del archivo comprimido (.bin).

Estructuras abstractas utilizadas:

- Vectores de bytes: Eficientes para manejar datos binarios completos.
- Buffers dispersos: Representan únicamente los bits relevantes del flujo.

Funciones principales:

- leerBytes()
- parseHeader()
- extraerDiccionario()

## 3. Flujo completo del sistema

1. Leer archivo .txt
2. Construir frecuencias
3. Generar árbol Huffman
4. Codificar texto
5. Guardar .bin
6. Leer .bin mediante Lector
7. Reconstruir diccionario
8. Decodificar bit a bit
9. Producir texto final

## 4. Ejemplo de uso

Diccionario: 0   A, 10   B, 11   C  
Entrada: 0110110  
Salida: ACA

## 5. Justificación del diseño

El uso de librerías independientes mejora la claridad y mantenibilidad del proyecto. Cada librería encapsula una responsabilidad: estructuras de códigos (Dictionary), algoritmo de compresión (Huffman), y lectura/parseo de binarios (Lector). El diseño se basa en estructuras dispersas, árboles y diccionarios hash para optimizar memoria y tiempo.