Angel Armas -24714

Julio Pellecer -24

**Informe: Compresor y Descompresor de Huffman**

**Estructura General del Programa**

El programa de compresión y descompresión utilizando el algoritmo de Huffman se ha implementado en Python utilizando el paradigma de programación orientado a objetos. La estructura general del programa es la siguiente:

1. **Clase HuffmanNode**: Representa un nodo en el árbol binario de Huffman, con propiedades para el carácter, frecuencia y referencias a los nodos hijos.
2. **Clase HuffmanCompressor**: Clase principal que implementa todas las funcionalidades necesarias:
   * Cálculo de frecuencias de caracteres
   * Construcción del árbol de Huffman
   * Generación de códigos para cada carácter
   * Serialización y deserialización del árbol
   * Compresión y descompresión de archivos
3. **Clase HuffmanTests**: Implementa pruebas unitarias completas para verificar el correcto funcionamiento de todas las funcionalidades.
4. **Función main()**: Proporciona una interfaz de usuario simple por consola para comprimir y descomprimir archivos.

**Decisiones de Diseño**

**Estructura de Datos**

1. **Árboles Binarios**: Implementados a través de la clase HuffmanNode con punteros al hijo izquierdo y derecho. Cada nodo contiene un carácter (solo en nodos hoja) y su frecuencia.
2. **Cola de Prioridad**: Utilizamos la biblioteca heapq de Python para implementar la cola de prioridad necesaria en la construcción del árbol de Huffman, garantizando que siempre se seleccionen los nodos con menor frecuencia.
3. **Diccionarios**: Empleados para almacenar las frecuencias de cada carácter (Counter) y los códigos binarios generados para cada carácter.

**Serialización del Árbol**

Para guardar el árbol de Huffman en un archivo .hufftree, se tomó la decisión de utilizar un formato de serialización basado en un recorrido preorder con marcadores:

1. Para cada nodo:
   * Si es una hoja, se guarda un marcador.
   * Si es un nodo interno, se guarda un marcador
2. Se utiliza la biblioteca pickle para persistir esta estructura, lo que facilita la recuperación posterior mientras mantiene el formato binario.

Esta decisión permite representar eficientemente la estructura del árbol y garantiza una reconstrucción exacta durante la descompresión.