CUCEI

CENTRO UNIVERSITARIO DE
CIENCIAS EXACTAS E INGENIERÍAS

UNIVERSIDAD DE
GUADALAJARA
Red Universitaria de Jalisco

División de Tecnologías para la Integración Ciber-Humana

Departamento de Ciencias Computacionales

Carrera: Ingeniería en Computación

Análisis de Algoritmos
Planeacion Proyecto Final
Participación

Profesor: Lopez Arce Delgado Jorge Ernesto
Quintero Arreola Laura Vanessa
Gutierrez Vazquez Axel
Hernandez Macias Axxel Gael
Sección: D06

Fecha de entrega: 14 de noviembre de 2025

Tabla de asignación de roles inicial

Equipo Amarillo:

Integrante	Tarea	Resultado	Fecha de entrega
Quintero Arreola Laura Vanessa	Realizar la introducción para el reporte.	Explicar el propósito del proyecto, su objetivo y la importancia de aplicar el algoritmo de Huffman en la compresión de canciones.	16 de noviembre
	Diseño algorítmico y selección de formato de entrada	Diseño del algoritmo con la implementación de Huffman así como tener claro el formato de entrada que se usará	16 de noviembre
	Implementar el algoritmo de Huffman	Código funcional que construya el árbol de Huffman, que permita comprimir y descomprimir correctamente los datos.	15 de noviembre
	Creación de código base	Programa que servirá como base para la implementación de la técnica Voraz	14 de noviembre
Gutierrez Vazquez Axel	Investigar información sobre el procesamiento de audio	Resumen con conceptos clave sobre cómo se representan los datos de audio y cómo pueden ser preparados para su compresión.	15 de noviembre
	Redactar el desarrollo del proyecto	Describir paso a paso cómo se llevó a cabo el trabajo, desde la planeación hasta las pruebas finales.	15 de noviembre
	Implementar el módulo de preprocesado de audio	Módulo que lea los archivos de audio, extraiga sus datos y los prepare para ser usados por el algoritmo de	15 de noviembre

		compresión.	
	Diseño y Creación de la interfaz gráfica	Interfaz sencilla e intuitiva que permita seleccionar canciones, ejecutar la compresión y mostrar resultados de manera visual.	14 de noviembre
Hernandez Macias Axxel Gael	Añadir técnica selecciona en el código	Añadirá la técnica que fue seleccionada al código base para completar el programa	15 de noviembre
	Hacer pruebas del funcionamiento y anotar los resultados.	Registro de las pruebas realizadas, comparando archivos originales y comprimidos, con notas sobre eficiencia y exactitud.	18 de noviembre
	Redactar el clímax y los resultados del proyecto en el reporte	Sección final del informe que presente los resultados obtenidos y la efectividad del algoritmo.	19 de noviembre
	Analizar la complejidad temporal	Calcular y Analizar la complejidad temporal del algoritmo	19 de noviembre

Plan de trabajo

Se llevarán a cabo 3 reuniones los días: 15, 17 y 19 de noviembre para comprobar avances del proyecto; El medio principal de comunicación será WhatsApp pero no se descarta la posibilidad de utilizar Meet para tener una mejor retroalimentación de los entregables.

Algoritmo seleccionado

Qué algoritmo voraz implementaremos

Implementaremos el algoritmo de Prim. Lo elegimos porque es corto de implementar, claramente voraz y fácil de justificar dentro del proyecto como algoritmo complementario a Huffman.

En qué parte del algoritmo base se integrará

Prim se integrará como un módulo de preprocesado opcional antes de aplicar Huffman. La idea: construir un grafo donde cada nodo representa un símbolo y las aristas pesan la “diferencia” entre símbolos

Metodología de implementación

1. Construcción del grafo: decidir el símbolo base, calcular peso entre pares.
2. Módulo Prim: implementar Prim con una cola de prioridad (min-heap) para obtener el MST sobre ese grafo.
3. Agrupado/orden: recorrer el MST para crear una ordenación o partición de símbolos (clusters de vecinos).
4. Preprocesado: aplicar delta-encoding o reasignación local de símbolos dentro de cada clúster.
5. Huffman: generar frecuencias sobre los nuevos símbolos/preprocesados y ejecutar el Huffman habitual.
6. Integración: añadir opción en la línea de comandos / GUI para activar/desactivar este preprocesado.
   Estructura modular: io_audio → graph_builder → prim_module → preprocessor → huffman_encoder → packager.

Descripción de GUI

La GUI contará con los siguientes botones: Seleccionar archivo, Comprimir, Descomprimir, mostrando mensajes de retroalimentación al usuario. Al comprimir o descomprimir un archivo, el programa mostrará botones con la opción de descarga del archivo correspondiente. Además, cuenta con la casilla(checkbox), que permite activar o desactivar el uso del algoritmo alternativo (Prim) durante el proceso.

Etapas

- **Análisis:** decidir símbolo base, definir métrica de peso entre símbolos, diseñar formato de cabecera y cómo serializar clusters/MST.
- **Codificación:** implementar graph_builder, prim_module (Prim), módulo de agrupado/delta, e integrar con el codificador Huffman ya existente.
- **Pruebas:** unit tests del módulo Prim (propiedades de MST), pruebas del preprocesado en ejemplos pequeños, pruebas end-to-end de compresión/descompresión.
- **Integración:** añadir opción en CLI/GUI, ejecutar pruebas automáticas y preparar entregables.

Validaciones

- **Correctitud de Prim:** tests unitarios que verifiquen que el resultado del módulo Prim es un árbol (nodos conectados).
- **No romper Huffman:** asegurar que al activar el preprocesado los archivos descomprimidos sean idénticos al original.
- **Medición de impacto:** para un conjunto de canciones de prueba comparar métricas con y sin Prim: tamaño original, tamaño comprimido, ratio de compresión y tiempo de ejecución.