**Estructura del Informe**

# Introducción

En esta sección se presenta una descripción general del reto, el objetivo del informe y la relevancia del uso de árboles binarios en la solución de problemas relacionados con estructuras jerárquicas.

# Capítulo 1 – Fase de Ideación

1. **Descripción del problema**Un grupo de arqueólogos ha descubierto los restos de una antigua civilización. Al analizar inscripciones y artefactos, descubren que la estructura social y familiar de dicha civilización puede representarse como un árbol genealógico. Para preservar esta información y estudiarla mejor, se necesita una aplicación que permita almacenar los datos familiares de forma jerárquica, eficiente y consultable.
2. **Requerimientos del sistema**

* Funcionales
  + Insertar miembro: agrega una nueva persona al árbol genealógico.
  + Buscar miembro: localiza un miembro por su ID.
  + Mostrar árbol inorden: muestra a todos los miembros en orden jerárquico (de menor a mayor ID).
  + Mostrar ancestros: muestra los antepasados de un miembro específico.
  + Mostrar descendientes: muestra los hijos y descendientes de un miembro.
  + Salir: cierra el programa.
* No funcionales
  + Buena organización del código: modularidad, funciones claras y uso de punteros.
  + Rápido y eficiente: las operaciones de inserción, búsqueda y recorrido se ejecutan rápidamente gracias al uso del Árbol Binario de Búsqueda.
  + Manejo básico de errores: el sistema detecta si se quiere insertar un ID duplicado o buscar un miembro inexistente.

1. **Respuesta a las preguntas guías**
   1. ¿Qué información se debe almacenar en cada nodo del árbol?
      1. el árbol genealógico se guarda como un "nodo", y ese nodo debe tener:
         1. Un **ID**
         2. **nombre** del miembro
         3. **Fecha de nacimiento**
         4. **Relación familiar** (padre, madre, hijo, hija).
         5. Dos punteros: uno a un nodo **izquierdo** y otro al **nodo derecho**, que permiten conectar a los demás miembros del árbol según las reglas del Árbol Binario de Búsqueda.
   2. ¿Cómo insertar y eliminar miembros del árbol sin romper su estructura?
      1. **Insertar**: se compara el nuevo ID con los ya existentes. Si es menor, se coloca a la izquierda; si es mayor, a la derecha. Así se mantiene el orden del árbol.
      2. **Eliminar**: Si el nodo no tiene hijos simplemente se borra. Si tiene un solo hijo se conecta el hijo directamente con el padre del nodo. Si tiene dos hijos se busca el siguiente nodo más cercano (sucesor) y se reemplaza.
   3. ¿Qué métodos permiten recorrer el árbol para visualizar la genealogía?
      1. **Inorden**: Recorre el árbol primero por izquierda, luego raíz, luego derecha. Mostrará los miembros **ordenados por ID.**
      2. **Preorden**: Recorre el árbol primero raíz, luego izquierda, luego derecha. Sirve para **ver cómo se arma el árbol desde arriba hacia abajo.**
      3. **Postorden**: Recorre el árbol primero izquierda, luego derecha, y al final la raíz. Se usa si quieres **eliminar todos los nodos desde abajo.**
   4. ¿Cómo determinar si un miembro pertenece a una rama específica?
      1. Para saber si alguien pertenece a una rama del árbol, se buscará por su ID o nombre usando una búsqueda normal en la teoría de los aboles de búsqueda binaria, una vez encontrado el nodo, se puede recorrer hacia arriba para ver quiénes son sus ancestros, o hacia abajo para ver sus descendientes, así podremos saber si ese nodo pertenece al árbol.
   5. ¿Cómo balancear el árbol si se vuelve demasiado profundo?
      1. Revisaremos la altura (o profundidad) de los subárboles izquierdo y derecho para verificar si el árbol está desbalanceado. Si la diferencia de altura entre ambos lados es significativa (mayor a 3), significa que el árbol está desbalanceado. En ese caso, podemos realizar una rotación manual para balancearlo y si es necesario, reconstruir el árbol creando una nueva estructura binaria más equilibrada.
2. **Herramienta Colaborativa**

# Capítulo 2 - Prototipo

1. **Descripción de estructuras de datos y operaciones:**
2. **Algoritmos principales:**

* *Pseudocódigo para crear un árbol binario.*
* *Pseudocódigo para realizar el recorrido de un árbol.*

1. **Diagramas de Flujo**
2. **Avance del código fuente**

# Capítulo 3 – Solución Final

1. **Código limpio, bien comentado y estructurado.**
2. **Capturas de pantalla de las ventanas de ejecución con las diversas pruebas de validación de datos**

# Capítulo 4 – Evidencias de Trabajo Colaborativo

1. **Repositorio con Control de Versiones (Capturas de Pantalla)**

* Registro de commits claros y significativos que evidencien aportes individuales (proactividad).
* Historial de ramas y fusiones si es aplicable.
* Evidencia por cada integrante del equipo.
* Enlace a la herramienta colaborativa

# Conclusiones

Reflexión sobre los aprendizajes alcanzados, dificultades enfrentadas durante la implementación, y posibles mejoras al sistema desarrollado.

# Referencias

Lista de fuentes consultadas (libros, artículos, sitios web), citadas según el formato ISO690 Numérico

# 8. Anexos (opcional)

Código completo, resultados adicionales, diagramas complementarios u otro material relevante.