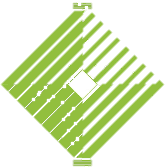
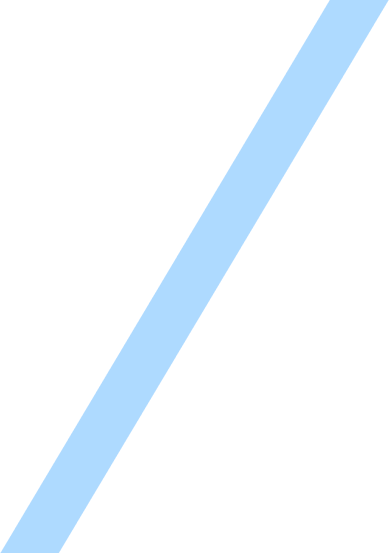
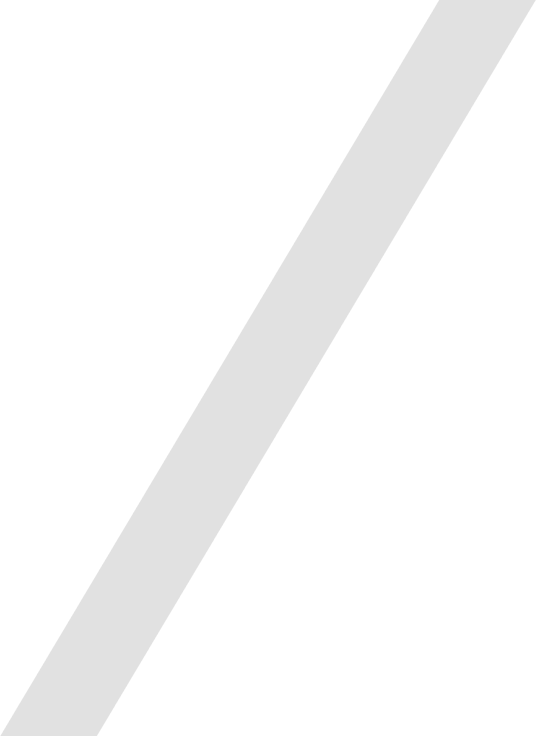
**DIVISIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN**

**CARRERA**

**• DESARROLLO DE SOFTWARE MULTIPLATAFORMA•**



*ACT1-1P-ÁRBOLES BINARIOS | ESTRUCTURA DE DATOS*

PROFESORA: MIRIAN MAGALY CANCHE CAAMAL

FECHA: 15 DE NOVIEMBRE DEL 2024

CUAT. - GRUPO: 4B-DSM

3ER PARCIAL

EDUARDO ALBERTO FLORES HERRERA

Un letrero de color blanco

Descripción generada automáticamente con confianza baja

Contenido

[INTRODUCCIÓN 3](#_Toc182609486)

[DESARROLLO 5](#_Toc182609487)

[TIPOS DE ÁRBOLES 5](#_Toc182609488)

[Recorridos en Árboles 7](#_Toc182609489)

[Herramientas, bibliotecas y propiedades para el manejo de árboles binarios en C#, Java y Python 8](#_Toc182609490)

[Ejemplos de Implementación de Árboles Binarios 14](#_Toc182609491)

[CONCLUSIÓN 20](#_Toc182609492)

[REFERENCIAS 21](#_Toc182609493)

## INTRODUCCIÓN

En esta práctica abordé el estudio de los árboles binarios como una estructura de datos fundamental en el desarrollo de software. Los árboles binarios permiten organizar y manejar información de manera eficiente, siendo útiles en diversos contextos como búsquedas, ordenación, y procesamiento de datos jerárquicos. A lo largo de esta actividad, investigué los tipos de árboles, detallando cada uno con ejemplos visuales y analizando los diferentes métodos de recorrido de los árboles binarios. Además, se exploraron herramientas y bibliotecas en lenguajes como C#, Java, y Python que facilitan la implementación y manejo de estructuras de árbol. Esta investigación y sus aplicaciones prácticas refuerzan la importancia de los árboles en la resolución de problemas tecnológicos.

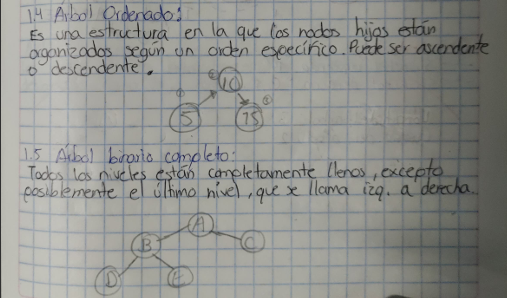
## DESARROLLO

## TIPOS DE ÁRBOLES

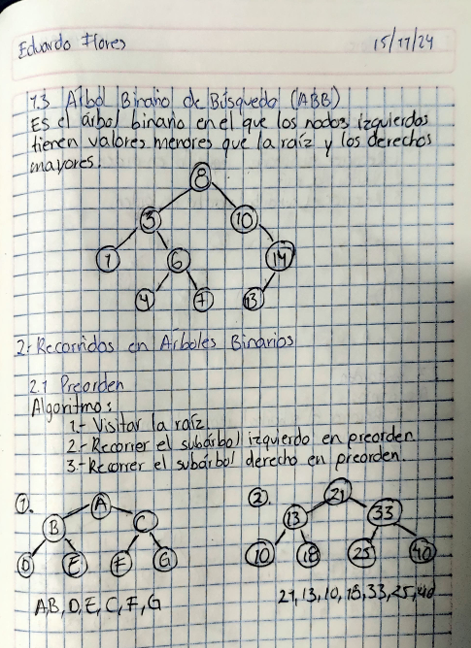
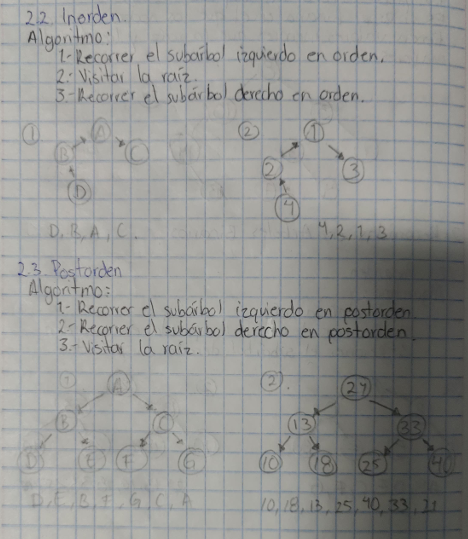
Diagrama

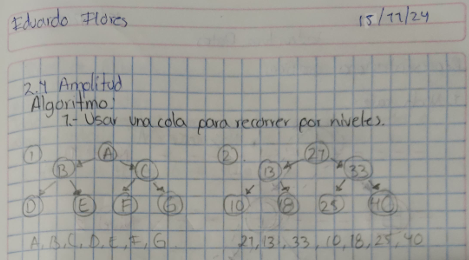
Descripción generada automáticamente

Gráfico

Descripción generada automáticamente

## Recorridos en Árboles



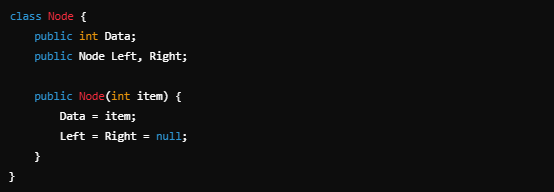


## Herramientas, bibliotecas y propiedades para el manejo de árboles binarios en C#, Java y Python

**1. C#**

**Herramientas:**

1. **Clases personalizadas**:
   * Permiten crear nodos y árboles binarios desde cero.
   * Cada nodo es una instancia de una clase que contiene al menos tres propiedades:
     + **Data**: Para almacenar el valor del nodo.
     + **Left**: Referencia al subárbol izquierdo.
     + **Right**: Referencia al subárbol derecho.  
       Ejemplo básico de una clase nodo:



1. **LINQ (Language Integrated Query)**:

* Herramienta para realizar consultas eficientes en árboles binarios que contienen datos.  
  Ejemplo: Filtrar nodos que cumplan una condición específica.



**Bibliotecas:**

1. **System.Collections.Generic**:
   * Proporciona colecciones genéricas como listas (List<T>), colas (Queue<T>) y pilas (Stack<T>), que son fundamentales para recorrer árboles.
   * Ejemplo: Recorrido por niveles usando una cola:

Texto

Descripción generada automáticamente

**Propiedades destacadas:**

* Permite implementar árboles personalizados mediante clases y nodos (como TreeNode).
* Soporte para métodos recursivos para recorridos (inorden, preorden, postorden).
* Integración con LINQ para consultas y filtrado de datos en los árboles.

**2. Java**

**Herramientas:**

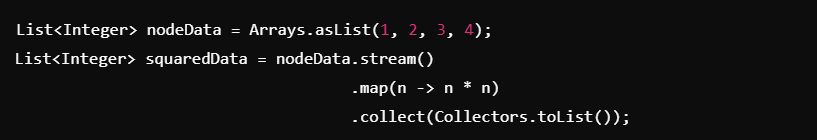
1. **Clases personalizadas**:
   * Los árboles binarios en Java también suelen implementarse mediante clases específicas que definen la estructura de un nodo y los métodos asociados.  
     Ejemplo básico de clase nodo:

Imagen que contiene Texto

Descripción generada automáticamente

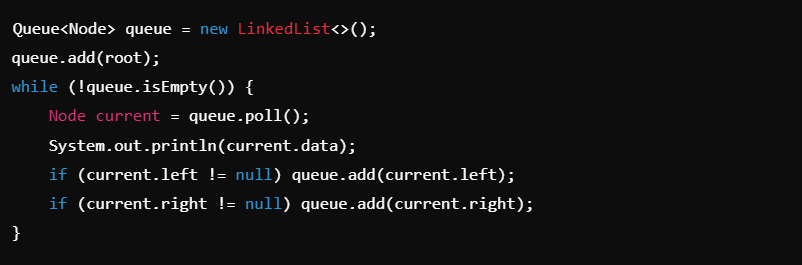
1. **Streams y Lambdas (desde Java 8)**:

* Simplifican operaciones sobre colecciones de nodos.  
  Ejemplo: Aplicar una transformación a una lista de datos:



**Bibliotecas:**

1. **java.util**:
   * Proporciona estructuras como:
     + **Queue**: Para implementar recorridos en amplitud (BFS).
     + **Stack**: Para búsquedas en profundidad (DFS).  
       Ejemplo: Uso de Queue para recorrido por niveles:



1. **Apache Commons Collections**:

* Biblioteca externa que incluye árboles binarios equilibrados (como AVL o Red-Black Trees).  
  Ejemplo de uso:

Texto

Descripción generada automáticamente

**Propiedades destacadas:**

* Uso de interfaces como Comparable para ordenar elementos en los nodos.
* Métodos eficientes para insertar, buscar y eliminar nodos en árboles binarios.
* Soporte para Java Streams y Lambdas en operaciones sobre árboles.

**3. Python**

**Herramientas:**

1. **Clases personalizadas**:
   * Permiten una implementación dinámica y sencilla de nodos y árboles.  
     Ejemplo básico:

Forma

Descripción generada automáticamente con confianza media

1. **Listas y Diccionarios**:

* Pueden simular estructuras jerárquicas como árboles.  
  Ejemplo: Representación de un árbol como diccionario:

Texto

Descripción generada automáticamente

**Bibliotecas:**

1. **collections**:
   * Contiene estructuras como deque (doble cola), que son útiles para implementaciones como recorridos por nivel.  
     Ejemplo:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **heapq**:

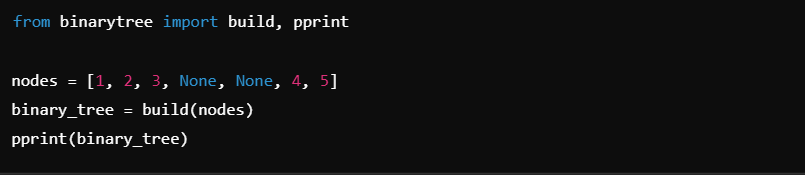
* Diseñada para trabajar con montículos binarios (estructuras especializadas de árboles binarios).  
  Ejemplo:

Texto

Descripción generada automáticamente con confianza baja

1. **binarytree** (biblioteca externa):

* Permite generar árboles binarios de manera visual y funcional.  
  Ejemplo:



**Propiedades destacadas:**

* Uso de listas anidadas o clases personalizadas para representar nodos.
* Soporte nativo para decoradores y funciones recursivas.
* Extensibilidad mediante librerías especializadas.

## Ejemplos de Implementación de Árboles Binarios

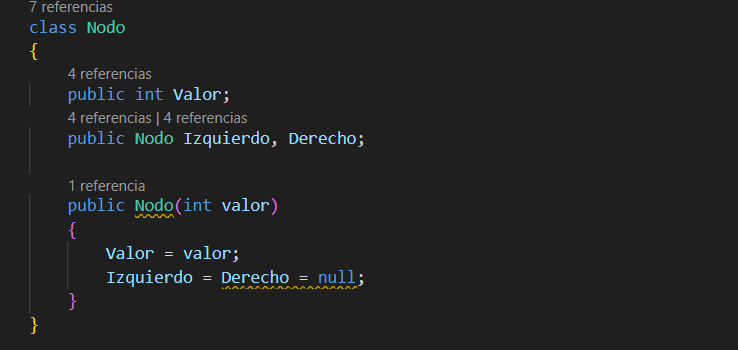
**Caso 1: Búsqueda de Rutas Óptimas (C#)**

**Descripción:**  
Implementación de un Árbol Binario de Búsqueda (ABB) para encontrar rutas óptimas en un sistema de navegación. Cada nodo representa una ubicación con un peso que indica el costo de moverse a esa ubicación.

**Código:**

Texto

Descripción generada automáticamente

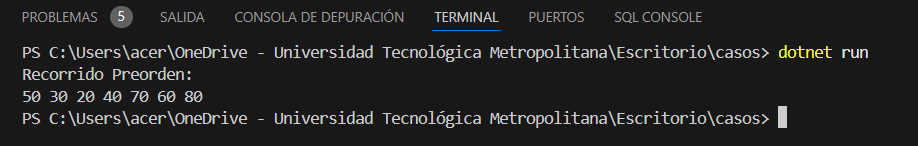


Texto

Descripción generada automáticamente

**Caso de prueba:**

* **Entrada:** Insertar los valores [50, 30, 70, 20, 40, 60, 80].
* **Resultado esperado:** Recorrido Preorden → 50 30 20 40 70 60 80.

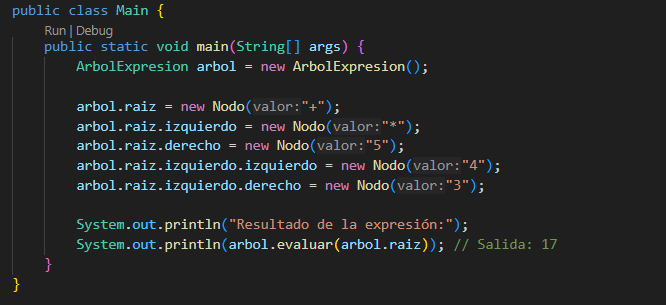


**Caso 2: Evaluación de Expresiones Matemáticas (Java)**

**Descripción:**  
Creación de un Árbol Binario para evaluar expresiones matemáticas. Los nodos hoja contienen números, y los nodos internos contienen operadores.

**Código:**

Texto

Descripción generada automáticamente

**Caso de prueba:**

* **Entrada:** Expresión ((4 \* 3) + 5).
* **Resultado esperado:** 17.

Texto

Descripción generada automáticamente

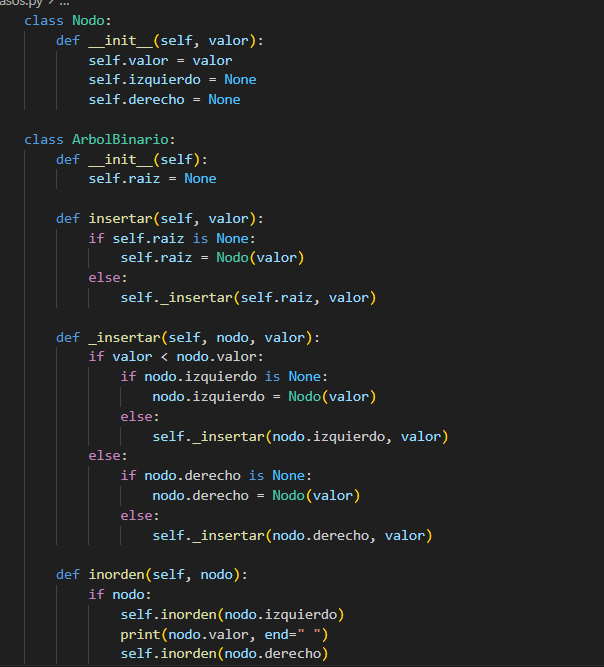
**Caso 3: Organización de Directorios (Python)**

**Descripción:**  
Uso de un Árbol Binario para simular un sistema de organización de directorios. Cada nodo representa un archivo o carpeta.

**Código:**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente



**Caso de prueba:**

* **Entrada:** Insertar ["Documentos", "Fotos", "Videos", "Musica"].
* **Resultado esperado:** Documentos Fotos Musica Videos.

Texto

Descripción generada automáticamente

## CONCLUSIÓN

La realización de esta práctica sobre **árboles binarios** me permitió entender en profundidad tanto la teoría como la implementación práctica de este tipo de estructuras de datos. Aprendí sobre los diferentes recorridos en profundidad (preorden, inorden, postorden) y en amplitud, y cómo se pueden utilizar en diversos lenguajes de programación. Además, comprender las aplicaciones reales de los árboles binarios en la optimización de búsquedas y la organización eficiente de la información me ha proporcionado una perspectiva clara sobre la relevancia de este conocimiento en el desarrollo de software. Esta práctica fue fundamental para fortalecer mis habilidades tanto conceptuales como prácticas en el área de estructuras de datos.

# REFERENCIAS

Programación y algoritmos. (n.d.). *Estructuras de datos: Árboles*. Recuperado de <https://www.programacionyalgo.com>

GeeksforGeeks. (n.d.). *Tipos y recorridos de árboles*. Recuperado de <https://www.geeksforgeeks.org>

Tecnología Fácil. (n.d.). *Estructuras de datos: Árbol binario*. Recuperado de <https://tecnologiafacil.com>

Aprende con expertos. (n.d.). *Recorridos de árboles binarios*. Recuperado de <https://www.aprendeconexpertos.com>

Tutoriales IT. (n.d.). *Árboles binarios en programación*. Recuperado de <https://www.tutorialesit.com>