

UNIVERSIDAD ESTATAL A DISTANCIA  
VICERRECTORÍA ACADÉMICA ESCUELA DE  
CIENCIAS EXACTAS Y NATURALES CARRERA  
INGENIERÍA INFORMÁTICA



CÁTEDRA INGENIERÍA DE SOFTWARE

ASIGNATURA

00825 ESTRUCTURA DE DATOS

PROYECTO 3

VALOR: 3.0

II CUATRIMESTRE 2025

# Estructura de Datos- Proyecto 3

## I Objetivo de aprendizaje

- Comprender la estructura, propiedades y tipos de árboles binarios.
- Implementar árboles binarios en Java, utilizando clases y punteros/referencias para crear nodos, enlazarlos, e insertar datos de forma recursiva o iterativa.
- Aplicar algoritmos de recorrido en árboles binarios: preorden, inorden, postorden y por niveles (BFS), e interpretar sus resultados.
- Analizar y resolver problemas utilizando árboles binarios, identificando su aplicabilidad para organizar, buscar y procesar información jerárquica.
- Evaluar la eficiencia de operaciones sobre árboles, como inserción, búsqueda y eliminación, en términos de complejidad temporal.

## II Objetivos de la actividad académica

- Evaluar la capacidad del estudiante para declarar y construir árboles binarios mediante la creación de nodos y enlaces entre ellos en Java.
- Verificar que el estudiante puede realizar correctamente los recorridos clásicos sobre árboles binarios (preorden, inorden, postorden y BFS).
- Evaluar la habilidad para resolver problemas prácticos utilizando árboles binarios de búsqueda, como búsqueda de elementos, ordenamiento implícito, inserciones y eliminaciones.
- Fomentar la aplicación de recursividad en estructuras jerárquicas, especialmente en operaciones sobre árboles.

## III Temas de estudio

- Árboles.
- Árboles binarios de búsqueda.

## IV Lineamientos generales

- Como herramienta de desarrollo se deberá utilizar el Netbeans en su versión más reciente. Se debe descargar el último release de la página [Apache Netbeans](#).
- También se debe utilizar el Java JDK más reciente. Pueden descargarlo de la página [Oracle](#).

- El producto por entregar corresponde al proyecto de Netbeans creado, incluyendo todas las librerías necesarias para su correcta ejecución. Este proyecto se debe subir a la plataforma Moodle en un archivo comprimido (.zip) con el nombre del estudiante y nombre del proyecto, antes de la fecha y hora límite de entrega.
- El programa debe realizarse en modo gráfico (GUI), es decir, no se permite en modo consola.
- No se permite el uso de cuadros de diálogo tipo MessageBox para solicitar o para mostrar datos. Para ello se pueden utilizar cajas de texto, etiquetas o listas gráficas según sea. La única excepción es para mostrar excepciones de la aplicación o para dar un mensaje al usuario por operaciones incorrectas en el sistema resultado de validaciones. Si la operación se realiza exitosamente, no se deben mostrar mensajes utilizando estos cuadros de diálogo.
- Los datos deben persistir en memoria en todo momento hasta que se cierre la aplicación.

## V Instrucciones metodológicas

- Tipo de actividad: Individual.
- Modalidad: virtual.
- Fecha límite de entrega:
- No se permite el uso de inteligencia artificial ni colaboración externa, salvo autorización expresa del docente.
- Desarrolla tus soluciones utilizando buenas prácticas de programación (claridad, indentación, comentarios donde sean necesarios).
- Asegúrate de que el código compile y funcione correctamente.

## VI Estructura del trabajo

Se debe crear un programa que gestione un árbol binario.

El árbol debe crearse con clases Nodo, donde cada objeto contendrá la información requerida y el puntero al siguiente elemento.

No se podrá utilizar clases ya implementadas de Java para la gestión de este árbol binario.

En una única pantalla, el usuario tendrá las siguientes opciones:

### *Inserción:*

El programa gestionará una colección de tarjetas de DC Comics.

En esta opción, el programa le solicitará al usuario los datos de una tarjeta de DC Comics que obtuvo de un sobre para incluirla en su colección, es decir, en el árbol binario. Los datos son:

Nombre del campo	Tipo de dato
Id de la tarjeta	Número entero (debe ser único)
Descripción de la tarjeta	Texto
Categoría de tarjeta	Se selecciona un valor de una lista (control tipo DropDownList) con las siguientes opciones: <ul style="list-style-type: none"><li>• Civiles</li><li>• Equipos</li><li>• Súper héroes</li><li>• Objetos</li><li>• Súper villanos</li><li>• Paneles</li><li>• Cara a cara</li><li>• Frases icónicas</li><li>• Película especial</li></ul>

El usuario debe crear un nodo con el objeto “Tarjeta” (basado en los campos anteriores) y el programa lo guardará en el árbol.

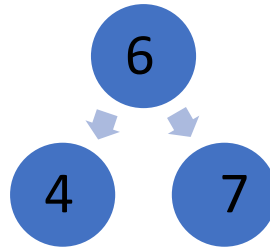
La clase Tarjeta será la siguiente:

```
public class Tarjeta
{
    private int Id;
    private string Descripcion;
    private string Categoria;
}
```

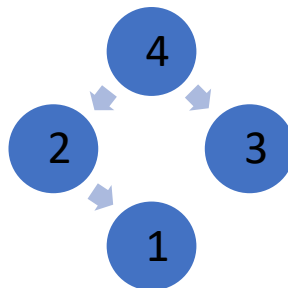
Antes de incluir el nodo, el programa debe buscar el valor del campo Id entre los nodos existentes en el árbol binario a ver si existe, ya que no se permitirá crear un nodo con un Id que ya fue incluido anteriormente.

El primer Nodo ingresado será la raíz. A partir del segundo Nodo a ingresar, se debe insertar en forma ordenada por el Id, de modo que, si el Id es menor al Id de la raíz, se debe insertar a la izquierda, de lo contrario a la derecha.

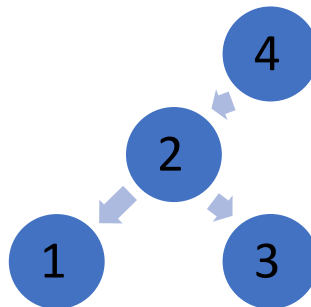
Ejemplo de árbol correctamente ordenado (el número mostrado es el del campo Id):



Ejemplo de árbol ordenado incorrectamente:



El ordenamiento correcto del árbol anterior es:



En el ejemplo anterior, para lograr ese ordenamiento, el primer elemento a insertar sería el que tiene el  $Id = 4$ , el cual queda de raíz. A partir de ahí se empieza a ordenar cada nuevo Nodo incluido. Posteriormente, ingresó el  $Id = 2$ , la cual al ser menor que el  $Id = 4$  se ubica a su izquierda. Luego ingresa el  $Id = 3$  el cual es menor que el  $Id$  de la raíz y mayor que el  $Id = 2$ , por lo que se ubica a la derecha del  $Id = 2$ . Posteriormente ingresa el  $Id = 1$ , el cual es menor que el  $Id = 4$  y menor que el  $Id = 2$ , entonces va a la izquierda de este.

No se deben insertar Nodos con el  $Id$  repetido. El programa deberá realizar la validación correspondiente recorriendo el árbol binario (no se permite utilizar otra estructura paralela). Si el valor del  $Id$  ya existe, se muestra un mensaje indicando que no se pudo incluir el elemento.

### *Eliminación:*

El usuario digita el valor del Id a eliminar. El programa procederá a eliminar el nodo correspondiente considerando lo siguiente:

1. Si el nodo es un nodo hoja, solo se elimina y el puntero de su padre se hace null.
2. Si el nodo tiene solo un subárbol hijo a su derecha, no se elimina y se muestra una etiqueta en la pantalla indicando que no se puede eliminar el nodo porque solo tiene un subárbol a su derecha.
3. Si el nodo tiene solo un subárbol hijo a su izquierda, se elimina y el padre del subárbol toma su lugar.
4. Si el nodo tiene dos subárboles hijos, no se elimina y se muestra una etiqueta en la pantalla indicando que no se puede eliminar el nodo porque tiene 2 hijos.
5. En cualquier caso, si el nodo a eliminar corresponde a una tarjeta de la categoría Civiles, no se podrá eliminar y deberá mostrar un mensaje al usuario indicando que no se permite eliminar tarjetas de Civiles.

### *Búsqueda de Id de una tarjeta*

El usuario indicará el valor del Id a Buscar dentro del árbol. Si existe, se mostrará la información completa del nodo (todos sus campos).

Esta búsqueda se realizará haciendo un recorrido del árbol usando los punteros de los nodos. No se permite el uso de una estructura paralela.

### *Recorrido PRE-ORDEN*

Esta opción mostrará la lista de nodos (campos Id separados por guiones) utilizando un recorrido pre-orden (no usar un cuadro de diálogo).

### *Recorrido POST-ORDEN*

Esta opción mostrará la lista de nodos (campos Id separados por guiones) utilizando un recorrido post-orden (no usar un cuadro de diálogo).

### *Recorrido IN-ORDEN*

Esta opción mostrará la lista de nodos (campos Id separados por guiones) utilizando un recorrido in-orden (no usar un cuadro de diálogo).

### *Cantidad de nodos en el árbol*

Esta opción mostrará la cantidad de tarjetas cuya categoría sea "Súper héroes" o "Súper villanos" del árbol (no usar un cuadro de diálogo).

### *Listar nodos hoja en el árbol*

Esta opción mostrará una lista de las descripciones de las tarjetas cuya categoría sea "Frases icónicas" y que adicionalmente sean nodos hoja del árbol (no usar un cuadro de diálogo).

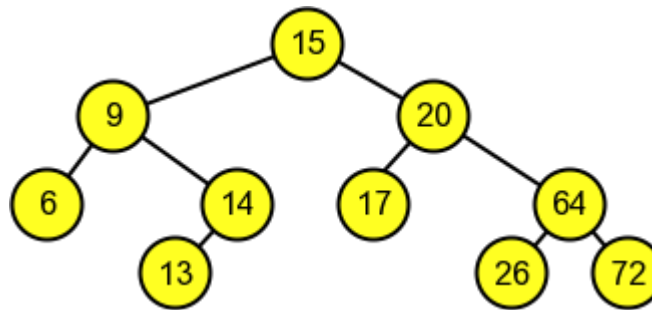
### *Tarjeta con el mayor y el menor id*

Esta opción mostrará los nodos del árbol que correspondan al mayor y al menor id (no usar un cuadro de diálogo). Ambos valores se obtienen haciendo un recorrido del árbol.

### *Graficar el árbol en pantalla*

En esta opción, se mostrará en pantalla el dibujo del árbol creado, mostrando solo la información de los Id. Debe ser utilizando gráficos o formas de cualquier librería de java.

Ejemplo de árbol dibujado o graficado en pantalla:



\* Si las librerías requeridas no se incluyen en el proyecto subido a la plataforma Moodle, el estudiante perderá los puntos de esta funcionalidad según la rúbrica.

Cada una de las funciones se pueden realizar en cualquier momento y en cualquier orden.

No se debe limpiar el árbol binario en ningún momento, a menos que sea utilizando la función de Eliminación (para todos los nodos) o al detener la ejecución del programa.

## VII Rúbrica o escala de evaluación

NO.	INDICADORES POR EVALUAR	CUMPLIMIENTO		PUNTOS
		Cumple	No cumple	
INSERCIÓN				
1.	El programa le solicita al usuario los datos de una tarjeta coleccionable incluyendo la información de los 3 campos especificados en el apartado “Inserción” del enunciado, considerando sus respectivos tipos de dato.			3
2.	El programa crea un árbol binario con clases Nodo, las cuales incluyen los 3 campos indicados en el enunciado, así como el puntero al siguiente elemento. No utiliza clases ya implementadas de Java para la gestión del árbol binario. La inserción se hace considerando el ordenamiento indicado en el enunciado en el apartado Inserción.			3
3.	No se permite la inserción de nodos con el Id repetido. El programa realiza la validación correspondiente recorriendo el árbol binario (no utiliza otra estructura paralela). Si el valor del Id ya existe, se muestra un mensaje indicando que no se pudo incluir el elemento.			1
ELIMINACIÓN				
4.	El programa elimina correctamente nodos hojas.			1
5.	El programa elimina correctamente nodos con un solo hijo o subárbol a la izquierda.			1
6.	El programa valida si el nodo a eliminar tiene un solo subárbol a su derecha e impide su eliminación mostrando el mensaje correspondiente.			1
7.	El programa valida si el nodo a eliminar tiene dos hijos e impide su eliminación mostrando el mensaje correspondiente.			1
8.	El programa valida si el nodo a eliminar corresponde a una tarjeta de la categoría Civiles e impide su eliminación mostrando el mensaje correspondiente.			1
BÚSQUEDA DEL ID DE UNA TARJETA				
9.	El usuario puede indicar el valor del Id a Buscar dentro del árbol. Si el nodo a buscar existe, se mostrará la información completa del nodo incluyendo todos sus campos. Esta búsqueda se realizará haciendo un recorrido del árbol usando los punteros de los nodos. No se permite el uso de una estructura paralela.			3
RECORRIDOS Y OTROS				
10.	El programa hace un recorrido PRE-ORDEN del árbol y muestra el resultado (lista de IDs separados por guiones) sin usar un cuadro de diálogo (por ejemplo, el MessageBox).			1
11.	El programa hace un recorrido POST-ORDEN del árbol y muestra el resultado (lista de IDs separados por			1



	guiones) sin usar un cuadro de diálogo (por ejemplo, el MessageBox).			
12.	El programa hace un recorrido IN-ORDEN del árbol y muestra el resultado (lista de IDs separados por guiones) sin usar un cuadro de diálogo (por ejemplo, el MessageBox).			1
13.	El programa muestra la cantidad total de tarjetas cuya categoría sea "Súper héroes" o "Súper villanos" del árbol (no usa un cuadro de diálogo).			1
14.	El programa muestra una lista de las descripciones de las tarjetas cuya categoría es "Frases icónicas" y que adicionalmente correspondan a nodos hoja del árbol (no usa un cuadro de diálogo).			1
15.	El programa muestra el nodo del árbol que tiene el mayor valor, entendiendo que es mayor un nodo cuyo precio es 4000 y la cantidad es 2, que otro con el precio de 7000 y la cantidad es 1.			1
16.	El programa muestra los nodos del árbol que corresponden al mayor y al menor id (sin usar un cuadro de diálogo). Ambos valores se obtienen haciendo un recorrido del árbol.			1
<b>GRAFICAR EL ÁRBOL EN PANTALLA</b>				
17.	Se muestra en pantalla el dibujo del árbol creado. Debe ser utilizando gráficos o formas de cualquier librería de java. Si las librerías requeridas no se incluyen en el proyecto subido a la plataforma Moodle, el estudiante perderá todos los puntos. Si solo se muestra el contenido del árbol sin graficarlo en pantalla tal como el ejemplo indicado en el enunciado, se perderán todos los puntos.			12
<b>GENERALIDADES</b>				
18.	Se utiliza el modo gráfico para todas las funcionalidades del enunciado, sin excepción.			5
19.	No utiliza cuadros de diálogo (tipo MessageBox) para dar ni para solicitar información al usuario ni para operaciones exitosas, a menos que sean excepciones del programa o que el usuario omitió algo necesario para la operación que quiere realizar.			1
20.	Cada una de las funciones se pueden realizar en cualquier momento y en cualquier orden.			5
21.	No se limpia o reinicia el árbol binario en ningún momento, a menos que sea utilizando la función de Eliminación (para todos los nodos) o al detener la ejecución del programa.			5