
	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN</p> <p style="text-align: center;">FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</p> <p style="text-align: center;">ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-002</p>	<p>Página: 1</p>

INFORME DE LABORATORIO
(formato de estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Estructura de Datos y Algoritmos				
TÍTULO DE LA PRÁCTICA:	Detector de Plagio				
ENLACE DEL REPOSITORIO	https://github.com/ianthony4/EDA-LAB-07-08				
NÚMERO DEPRÁCTICA	07 - 08	AÑO LECTIVO:	2023 - A	NRO. SEMESTRE	III
FECHA DE PRESENTACIÓN	07/08/2023		HORA DE PRE-SENTACIÓN:	18:30	
INTEGRANTE (s): <ul style="list-style-type: none">CHAISA FERNANDEZ, ANTHONY LEONELCORDOVA SELLERICO, JAMES SEUSLLACMA QUISPE, KEVIN ANDREE				NOTA:	Pendiente
DOCENTE (s): <ul style="list-style-type: none">Mg. Edith Giovanna Cano MamaniMg. Richart Smith Escobedo QuispeDra. Karim Guevara Puente de la Vega					

SOLUCIÓN Y RESULTADOS
I. METODOLOGÍA
<p>Estructura de Datos : Como estructura de datos hemos escogido un árbol AVL por las siguientes características.</p> <ul style="list-style-type: none"> • La propiedad de balanceo garantiza que la altura del árbol sea de $O(\log n)$. • En cada nodo del árbol se guarda información de la altura. • La altura del árbol vacío es -1. • Al realizar operaciones de inserción/eliminación se debe actualizar/balancear si fuera necesario.

II. CLASES UTILIZADAS

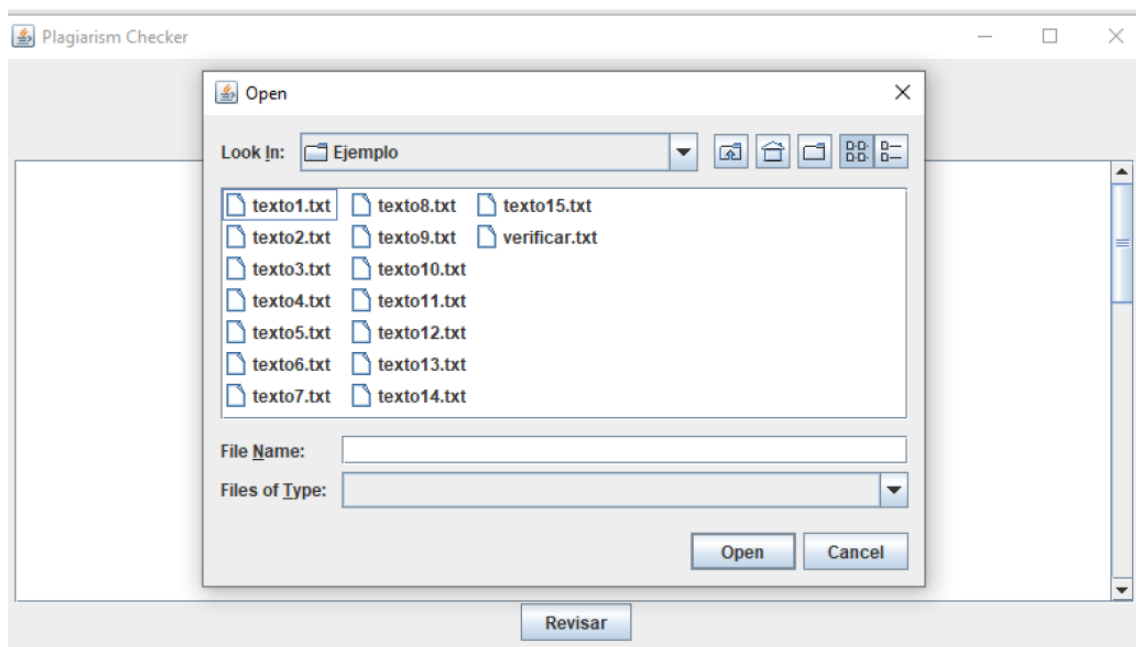
- **Clase Node** : Estructura del nodo del árbol AVL (Cada nodo contendrá un grupo de 10 palabras)
- **Clase AVLTree** : Implementación del árbol AVL (Se implementan métodos de inserción, balance, rotación a la derecha, rotación a la izquierda)
- **Clase Phrase** : Esta clase almacena un grupo de 10 palabras en su atributo "Data", luego estos grupos de 10 palabras mediante el método "addword" se usaran para ser insertados en el árbol AVL de la base de datos, también se usa el método "addword" para separar en grupos de 10 palabras el texto a ser comparado.
- **Clase Document** : En esta clase posee como atributos nombre "fileName" y el grupo de 10 palabras "phase"; además posee el métodos "createAVL()" el cual nos permite crear los árboles de la base de datos; también tenemos el método "matching" el cual realiza la búsqueda del un frase de 10 palabras en los diferentes árboles de la base de datos.
- **Clase PlagiarismChecker** : Esta clase implementa el método "LoadFiles(String[] paths)" el cual nos permite cargar los documentos para la base y el documento a revisar, también tenemos el método "verifyPlagiarism(String path)" en el cual se hacen las verificaciones para corroborar si hay plagio
- **Clase ResultChecker** : Esta clase tiene como atributos un arreglo de booleanos con los resultados de las verificaciones.
- **Clase Interfaz** : Esta clase contiene la interfaz para cargar archivos y realizar la ejecución de las verificaciones.
- **Clase Main** : Esta clase inicia la ejecución de la interfaz.

III. FUNCIONAMIENTO

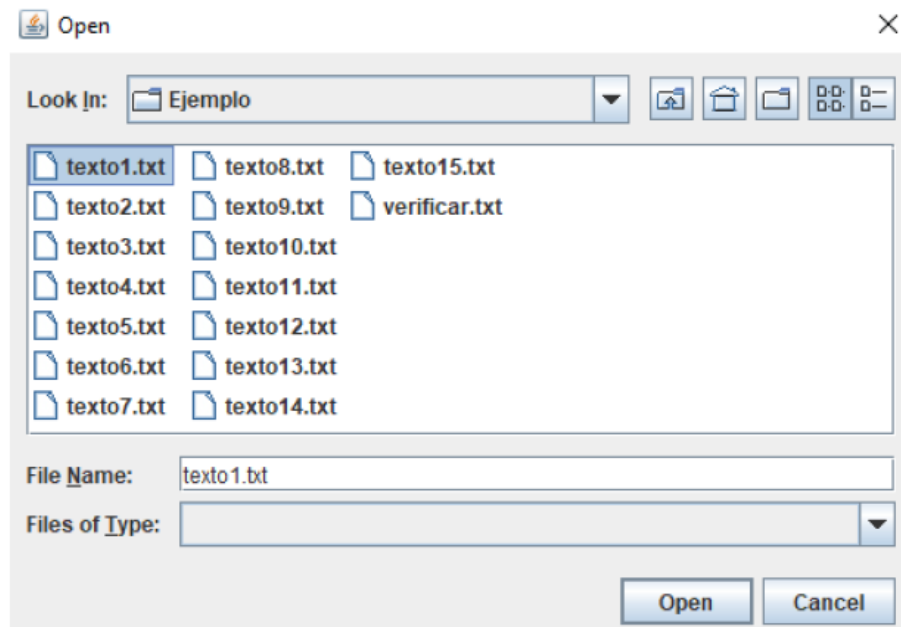
- **Interfaz Gráfica de Usuario (GUI)** : Al ejecutar el programa, se muestra una interfaz gráfica de usuario que permite al usuario interactuar con el detector de plagios. La interfaz contiene botones para cargar archivos en la base de datos, seleccionar un archivo para verificar el plagio y realizar la detección de plagio. También incluye un área de texto para mostrar el progreso y los resultados de la detección.
- **Carga de Archivos en la Base de Datos** : El usuario puede cargar varios archivos en la base de datos haciendo clic en el botón "Subir archivos a la Base de Datos". Cada archivo se agrega a la base de datos, y se crea un árbol AVL para cada archivo que permite una búsqueda rápida de palabras.
- **Selección de Archivo a Verificar** : El usuario puede seleccionar un archivo específico para verificar el plagio haciendo clic en el botón "Seleccionar archivo". El archivo seleccionado se comparará con los archivos en la base de datos para determinar si hay plagio.
- **Detección de Plagio** : Cuando se selecciona el archivo para verificar, el usuario puede hacer clic en el botón "Revisar" para iniciar la detección de plagio. El programa recorre el archivo seleccionado y crea una "frase" de palabras (un grupo de palabras contiguas) que contiene un máximo de 10 palabras. Luego, busca esa frase en cada árbol AVL correspondiente a los archivos en la base de datos. Si la frase se encuentra en algún árbol AVL, se considera un caso de plagio para ese archivo.
- **Resultados** : Después de completar la detección de plagio, el programa muestra los resultados en el área de texto de la interfaz gráfica. Muestra qué archivos en la base de datos tienen plagio con el archivo seleccionado y cuáles no. Los resultados se presentan de manera clara y ordenada.

IV. RESULTADOS DE PRUEBA

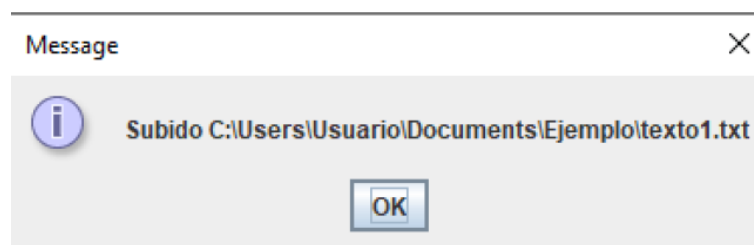
- Para los experimentos, nosotros subimos previamente varios archivos de texto a la base de datos de la carpeta Data, así como también el archivo que va a ser verificado, en este caso “verificar.txt”



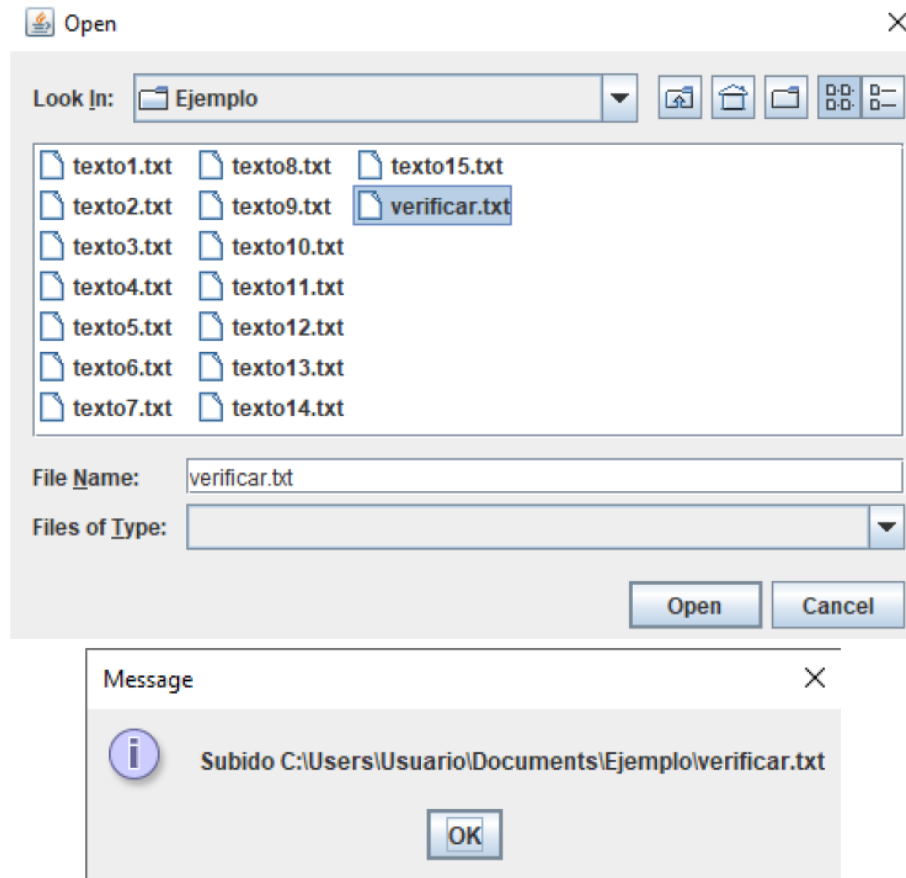
- Seleccionamos el archivo como fuente



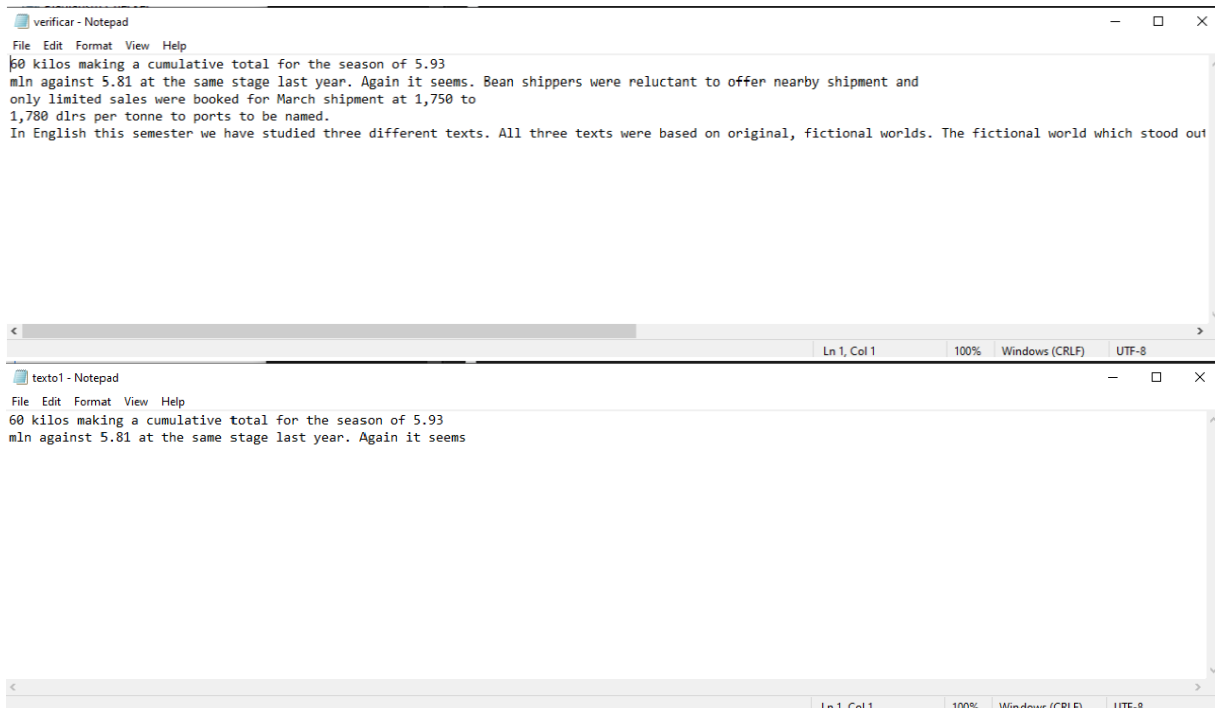
- Te saldrá un mensaje con la ruta donde extrajiste el archivo.

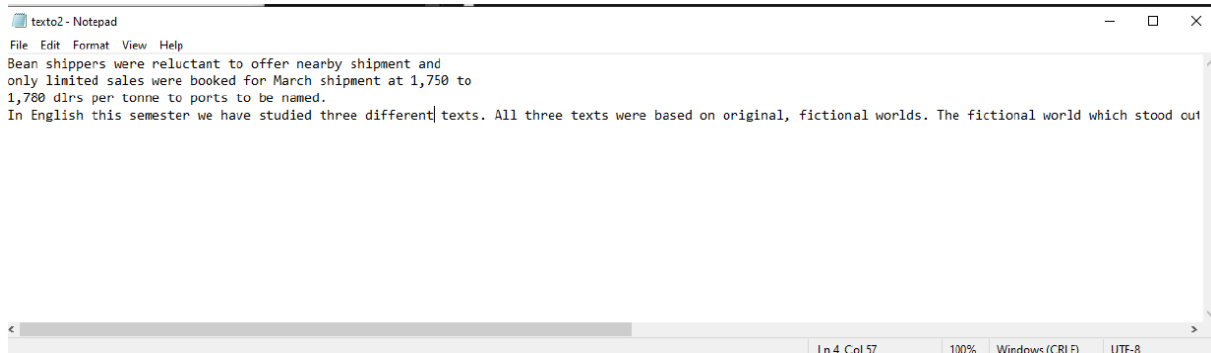


- Luego indicamos el archivo que analizaremos.

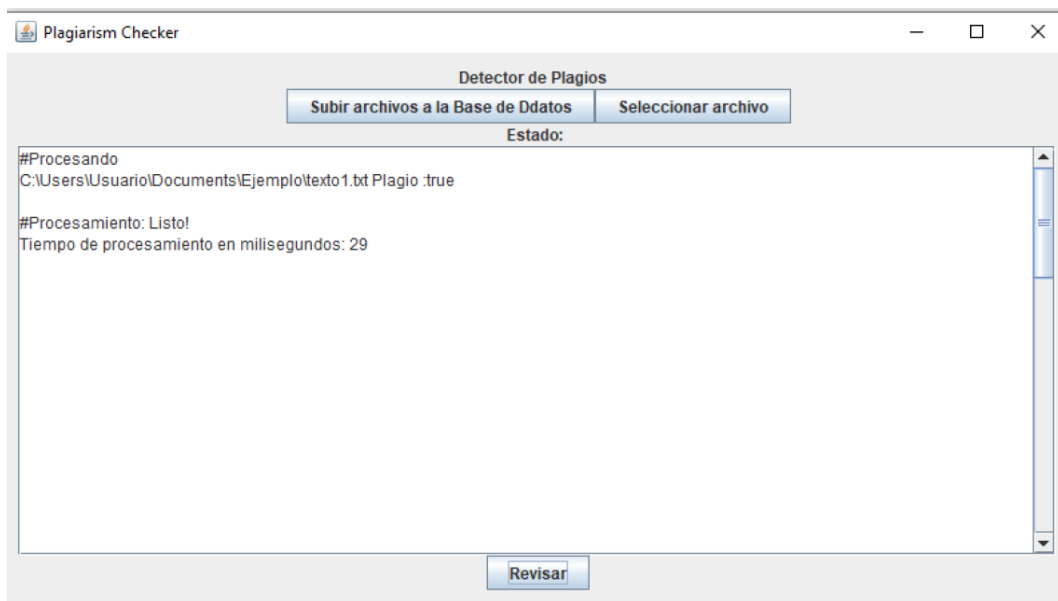


- En este caso nosotros, el contenido de “verificar.txt” tenía copiado a “texto1.txt” y “texto2.txt”.

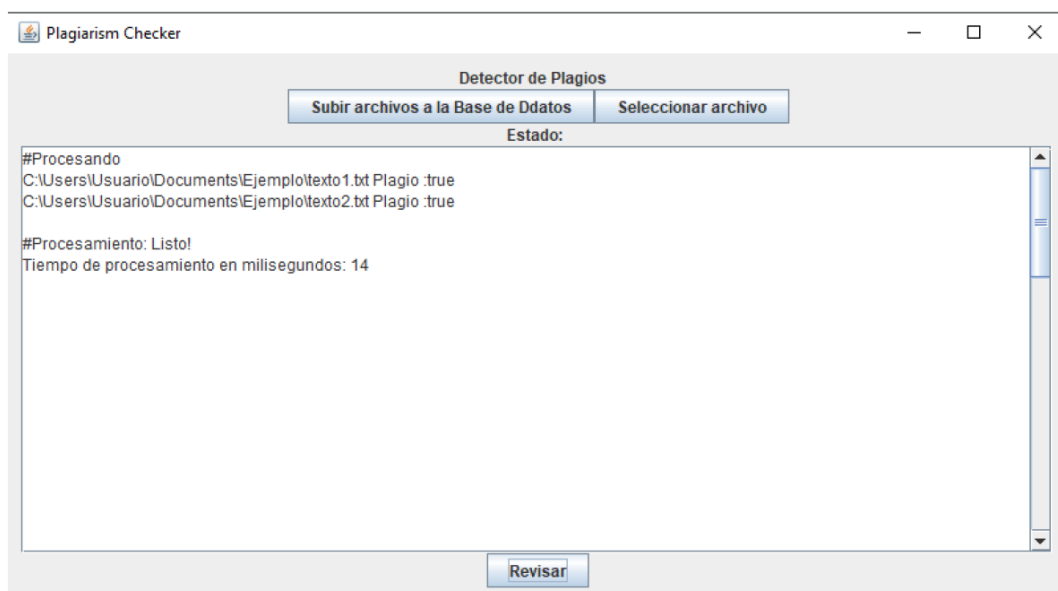




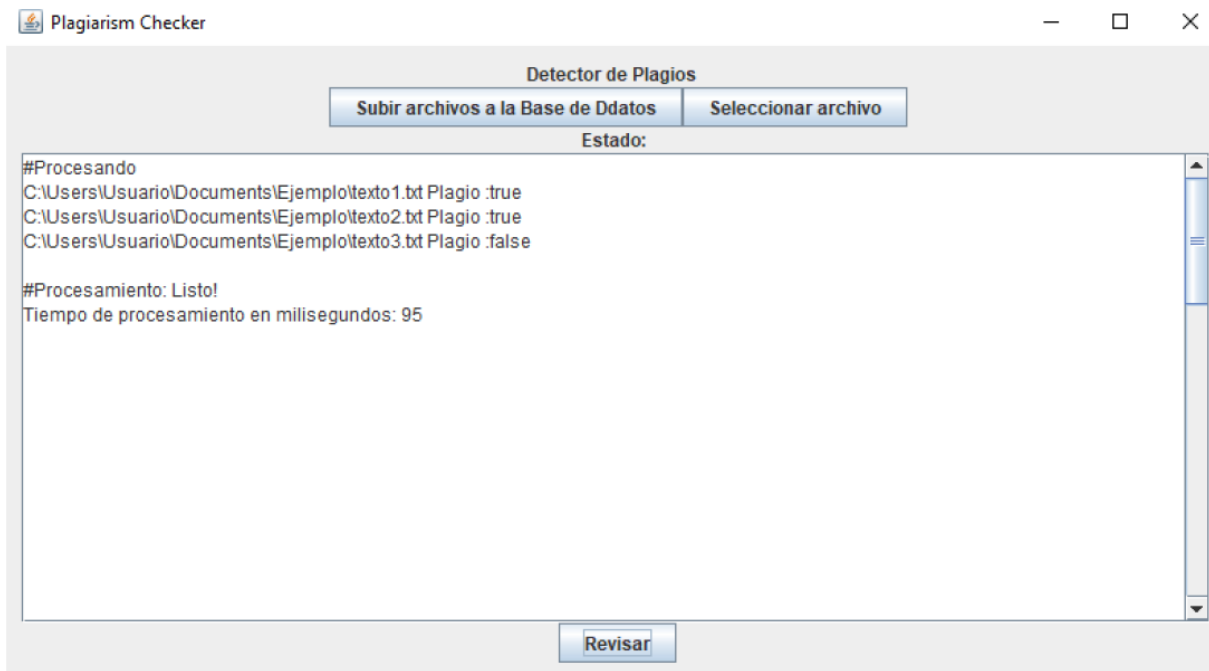
- Luego al darle comparar con el “texto1.txt” saldra el archivo del que tiene plagio y saldrá “true” si hay plagio y “false” en caso contrario.



- Ahora si queremos saber si “verficar.txt” tiene también plagio del “texto2.txt” o de “texto3.txt”. agregamos más fuentes. Hacemos el mismo procedimiento de subir archivo a la base de datos y le damos a revisar.



- Como vemos también sale “true” con respecto al “texto2.txt” quiere decir que también hay plagio de ese archivo. Ahora intentamos con el “texto3.txt”.



- Observamos que sale “false” quiere decir que no hay plagio de ese archivo.

V. CONCLUSIONES

- En el presente trabajo pudimos conocer más sobre el árbol AVL, ya que la implementamos como estructura para el almacenamiento de la base de datos.
- Como dificultad al desarrollar se tuvo la adaptación al modelo ya predeterminado, Clase Plagiarism-Checker y ResultChecker.

RETROALIMENTACIÓN GENERAL

...

REFERENCIAS Y BIBLIOGRAFÍA

- Pérez, G. M. C.-. (2016). Definición de los árboles AVL. AVL Definición. Recuperado 2023, de http://163.10.22.82/0AS/AVL_Definicion/index.html
- Javaz, Z. (s. f.). Diseño e implementación de un árbol binario balanceado (árbol AVL) de algoritmo y estructura de datos Java - programador clic. Programmerclick. Recuperado 02 de agosto de 2023, de <https://programmerclick.com/article/3400189243/>
- w3schools (s. f.). Java Tutorial. Recuperado 02 de agosto de 2023, de <https://www.w3schools.com/java/default.asp>
- Oracle. (s. f.). Java Platform SE 7. Recuperado 02 de agosto de 2023, de <https://docs.oracle.com/javase/7/docs/api/>
- <https://www.geeksforgeeks.org/tree-data-structure/?ref=lbp>