

**DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN**

**ESTRUCTURAS DE DATOS**

**Docente:**

**NRC:** 7174

Jimena Nayeli Tutillo Tutillo

Jorge Andrés Nasimba Pachacama

**Grupo 7**

**Octubre 2021-Marzo 2022**

1. **Proyecto: Traductor de palabras Español-Inglés, Inglés-Español.**
2. **Objetivos**

* Usar arboles avl para obtener las palabras con su respectiva traducción en un archivo txt previamente cargado.
* Crear un menú para dividir las opciones que se determinaron para el uso del traductor.

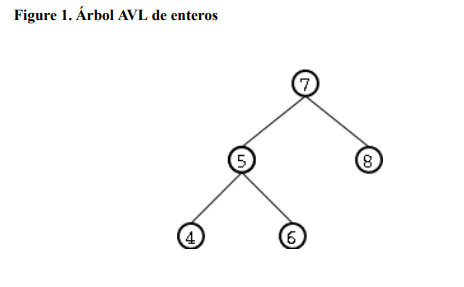
1. **Desarrollo**

Para la codificación del proyecto se realizó un menú principal para iniciar el traductor o salir, posteriormente un menú con 3 opciones las cuales se puede escoger moviendo las teclas direccionales y se selecciona la opción con un enter.

En la primera opción podemos buscar una palabra con su significado en inglés y español, sin embargo, antes de buscar una palabra debemos elegir si buscar la palabra en inglés o español.

En la segunda opción tenemos un menú similar al anterior donde podemos agregar, eliminar palabras en español-inglés, una opción para imprimir las palabras que se tienen en el traductor tanto en inglés como español y una opción para imprimir el árbol AVL. Se usan archivos txt para poder almacenar las palabras con sus respectivas traducciones en español e inglés, posteriormente se lee estos archivos para poder almacenar en el árbol AVL y poder optar por la búsqueda, agregación, eliminación, e impresión de palabras traducidas. Posteriormente realizamos se realiza un backup de las palabras en un archivo txt para evitar la pérdida de datos o el mal manejo y no inutilizar el programa.

Un árbol AVL es un árbol binario de búsqueda que cumple con la condición de que la diferencia entre las alturas de los subárboles de cada uno de sus nodos es, como mucho 1. La denominación de árbol AVL viene dada por los creadores de tal estructura (Adelson-Velskii y Landis). Recordamos que un árbol binario de búsqueda es un árbol binario en el cual cada nodo cumple con que todos los nodos de su subárbol izquierdo son menores que la raíz y todos los nodos del subárbol derecho son mayores que la raíz. Recordamos también que el tiempo de las operaciones sobre un árbol binario de búsqueda son O(log n) promedio, pero el peor caso es O(n), donde n es el número de elementos. La propiedad de equilibrio que debe cumplir un árbol para ser AVL asegura que la profundidad del árbol sea O(log(n)), por lo que las operaciones sobre estas estructuras no deberán recorrer mucho para hallar el elemento deseado. Como se verá, el tiempo de ejecución de las operaciónes sobre estos árboles es, a lo sumo O(log(n)) en el peor caso, donde n es la cantidad de elementos del árbol.



Algunas consideraciones sobre la implementación del tipo de dato abstracto

• Las declaraciones que se listarán a continuación no tienen por qué tomarse al pie de la letra. Cada programador tendrá su estilo y su forma de resolver sus problemas.

• Las declaraciones que se listarán a continuación no tienen por qué tomarse al pie de la letra. Cada programador tendrá su estilo y su forma de resolver sus problemas.

• Como se podrá ver en el siguiente listado, la única diferencia de los nodos de un árbol AVL con los de un árbol binario común es la variable altura en la estructura nodo. • Los nodos de un árbol pueden almacenar cualquier tipo de dato, arbitrariamente complejo. En este documento, por razones de simplicidad se usará el tipo de dato más simple que soporte comparaciones, o sea los enteros (tipo int de Ansi C). En el caso de que los datos almacenados en cada nodo sean más complicados (por ejemplo, estructuras) o sean dinámicamente almacenados en memoria, algunas funciones

**Bibliografía**

*http://es.tldp.org/Tutoriales/doc-programacion-arboles-avl/avl-trees.pdf*

*Estructuras de datos y algoritmos, sección 4.4, pág. 114, Mark Allen Weiss.*

*Data Structure Techniques, Standish, 1980, section 3.7.3.*

*Handbook of Algorithms and Data Structures, Gonnet, 1984, section 3.4.1*.