Trabajo Integrador - Propuesta de Investigación Cátedra de Programación I

Título: Estructuras de Datos Avanzadas.

Alumnos:
- **Justo Bellingi**
- **Matias Bejarano**

Fecha de entrega: 09/06/2025

Tutor: Martín A. García

Profesor: Cinthia Rigoni

Introduccion:

En este trabajo de investigacion, lo dividiremos en 4 partes. Una breve introduccion del tema, un marco teorico explicando los conceptos mas importantes relacionados a las estructuras de datos avanzados, un ejercicio practico que fue realizado en VsCode y subido a un repositorio de GitHub y por ultimo una conclusion sobre el tema y el trabajo.
Las estructuras de datos son una forma de organizar, procesar, recuperar y almacenar datos en una computadora.
Los usuarios pueden acceder y actualizar rápidamente la información organizada, lo que les permite manipular y utilizar los datos fácilmente.
Existen varios tipos de estructuras de datos básicas y avanzadas: la que elija una organización dependerá de los requisitos específicos del sistema.
Las estructuras de datos ayudan a crear sistemas de software robustos, eficientes y fáciles de mantener, esenciales para el rendimiento de una organización.
Constituyen la base de diversos algoritmos y técnicas de procesamiento, lo que permite la resolución eficaz de problemas en el ámbito del desarrollo informático y de software.

Ejemplos de arboles que se pueden ver en la vida real:

- Familias (Padres de tus padres,Padres, hijos)
- Paises (Nombre del pais, provincia, ciudad de esa provincia)
- Facultades (Directivos, profesores, alumnos)
- Clubes de futbol (Directiva, entrenador, jugadores)

Como se ve, si nos ponemos a pensar la jerarquía existe en todos lados. Desde la familia, la escuela o un club de fútbol, todo tiene un orden y una estructura, la cual si no existiese la vida tendria muy poca organizacion. Los árboles nos ayudan a entender cómo se relacionan las cosas. Sirven para organizar mejor y ver quién depende de quién.

Marco teorico:

Un árbol es una estructura de datos, que tiene varias utilizaciones que se pueden llevar a la vida real, como la organización de archivos en sistemas operativos, la gestión de bases de datos (por ejemplo, en árboles de búsqueda binaria), y en algoritmos de optimización.

Un árbol es una estructura de datos no lineal en la que cada nodo puede apuntar a uno o varios nodos. Son muy similares a las listas doblemente enlazadas en el sentido que tienen punteros que apuntan a otros elementos. Sin embargo, a diferencia de las listas doblemente enlazadas, no tienen una estructura lógica de tipo lineal o secuencial, sino ramificada.

Un grafo es una estructura matemática utilizada para representar relaciones entre objetos. Se compone de un conjunto de nodos (también llamados vértices) y un conjunto de aristas (también conocidas como enlaces o ramas).

Los árboles son representados, en general, a través de grafos, que tienen nodos conectados entre sí por arcos o ramas. Un nodo es una unidad básica de un árbol. Cada nodo contiene dos componentes principales: un valor y un puntero a sus nodos hijos, si es que los tiene. Los nodos en un árbol están conectados entre sí mediante estas referencias, formando una estructura jerárquica, tal como se muestra en la Figura 1.

Un árbol puede estar vacío y un árbol no vacío puede constar de un solo elemento (nodo raíz) o bien de varios elementos.

Clasificación de nodos según su ubicación en el árbol

Nodo raíz: es el punto de entrada a la estructura, generalmente ubicado en la parte más superior del árbol. No tiene un nodo padre y es único.

Nodo hoja: es cualquier nodo que no tenga hijos, es decir, un nodo terminal del árbol. Los nodos hoja están en los extremos de la estructura de un árbol y no tienen más ramificaciones debajo de ellos.

Nodo rama o interno: es cualquier nodo que tiene un padre y, al menos, un hijo. A diferencia de un nodo hoja, un nodo rama no es terminal y sigue propagando la jerarquía del árbol hacia abajo.

Clasificación de nodos según su relación con otros nodos

Nodo Padre: así como en una familia un padre es la persona que tiene hijos, un nodo padre es aquél que tiene uno o más nodos hijos conectados a él. Cada nodo, excepto la raíz, tiene solamente un nodo padre.

Nodo hijo: así como en una familia un hijo es una persona que tiene padres, un nodo hijo es aquél que está conectado con un nodo padre de modo que se encuentra directamente debajo de él en la jerarquía. El nodo hijo depende del nodo padre y no puede existir sin él.

Nodo hermano: así como en una familia los hermanos son hijos de un mismo padre, en un árbol los nodos hermanos son aquellos que comparten el mismo nodo padre. Es decir, tienen la misma distancia respecto al nodo raíz y a la vez están en el mismo nivel en la jerarquía del árbol.

Conjuntos anidados

Este tipo de representación utiliza la teoría de conjuntos, donde el nodo raíz es representado como el conjunto más abarcativo y sus nodos hijos son subconjuntos de éste.

Paréntesis anidados

En este tipo de representación, los árboles se describen mediante una notación en la que todos los nodos se encierran entre paréntesis. El paréntesis más externo representa el nodo raíz. Dentro de él se colocan sus nodos hijos y así sucesivamente hasta incluir todos los nodos del árbol.

(A (B (D, E) ,C (F)))

Indentación

Este tipo de representación utiliza indentaciones (sangrías) para reflejar la jerarquía de los nodos de manera visual, donde cada nivel de profundidad en el árbol se representa con una indentación adicional. Cada línea representa un nodo, y los nodos hijos se colocan en líneas subsiguientes con menor indentación. Este método es fácil de leer y comprender, especialmente cuando se trabaja con árboles simples en texto plano.

Propiedades de los árboles

Longitud = es el número de ramas o arcos que hay que transitar para llegar de un nodo a otro.

Profundidad = es la longitud de camino entre el nodo raíz y él mismo.

Nivel = es la longitud del camino que lo conecta al nodo raíz más uno. Un nivel puede contener uno o más nodos.

Altura = es el máximo nivel del mismo.

Grado = es el número de hijos que tiene dicho nodo. El grado de un árbol es el grado máximo de los nodos del árbol.

Orden = es la máxima cantidad de hijos que puede tener cada nodo.

Peso = es el número total de nodos que tiene un árbol. En programación es importante conocer el peso de un árbol dado que nos da una idea del tamaño del mismo

Árboles binarios
Es un árbol en el que cada nodo puede tener, como máximo, 2 hijos. Dicho de otra manera, es un árbol grado dos. En un árbol binario el nodo de la izquierda representa al hijo izquierdo y el nodo de la derecha representa al hijo derecho.

Formas de recorrer árboles binarios

Existen varias formas de recorrer un árbol. Las tres formas más comunes son preorden, inorden y postorden.

Preorden

El recorrido preorden es útil cuando necesitas realizar alguna operación en el nodo antes de procesar a sus hijos.

Inorden

El recorrido inorden es especialmente útil para los árboles de búsqueda binaria, ya que garantiza que los nodos se visiten en orden ascendente.

Postorden

El recorrido postorden es útil cuando quieres realizar alguna acción en los nodos hijos de un nodo antes de procesar el nodo mismo

---

Recursos utilizados:

Link del repositorio:
https://github.com/MOB-SD/UTN-TUPaD-Programaci-n-TPINT.git

Canva:
https://www.canva.com/design/DAGpTUVwTzE/VSYgIMH0ynyntWfqD7Kqig/edit?ut

m_content=DAGpTUVwTzE&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Video de youtube: https://youtu.be/8qh8hRjtVNs

Resultados:

Nuestro objetivo con este trabajo era hacer una estructura de arboles, donde cada uno pueda armar la estructura de datos que quiera. En nuestro caso en el video, armamos un ejemplo sobre un arbol de Argentina siendo el padre de Buenos Aires (puede ser el de cualquier provincia Argentina) y luego Buenos Aires siendo el padre de cualquier ciudad que pertenezca a Buenos aires y asi sucesivamente.

Conclusion:

Este trabajo nos dio a entender que las estructuras de datos, y en particular los árboles, que están presentes en muchísimos aspectos de la vida real. Pudimos ver cómo en los árboles se ven ejemplos de formas jerárquicas en contextos de la vida cotidiana. En el caso de la programacion se ve a traves de la jerarquia de nodos conectados por relaciones que se puede ver que se aplica igualmente en empresas, colegios y muchos lados mas.

Bibliografia:

https://docs.google.com/document/d/10k16oL15EeyOaq92aoi4qwK3t_22X29-FSV2iV-8N1U/edit?usp=sharing

https://amplitude.com/explore/data/what-how-data-structure