Árbol

Las estructuras de dato son utilizadas para almacenar y tener una excelente organización de diferentes tipos de datos. Podemos usar algoritmos para manipular y utilizar nuestras estructuras de datos. Existen diferentes tipos de estructuras, algunas más eficientes para un tipo de dato en específico.

Los arboles son estructuras de datos no lineales. Generalmente son utilizados para representar datos de manera jerárquica. Por ejemplo, una estructura jerárquica de una empresa, utiliza un árbol para organizarlo.

Los arboles son colecciones de nodos (vértices) y están vinculados con bordes o ramas, que representan las conexiones entre los nodos. Un nodo puede contener datos de cualquier tipo.

Usos de los arboles

Los arboles se pueden aplicar a una infinidad de cosas. Ya que la estructura jerárquica proporciona al árbol propiedades únicas para almacenar, manipular y obtener acceso a los datos. Como dato los arboles forman parte de la organización más básica de la computadora.

Almacenar como jerarquía: Almacenar información que se produce naturalmente en una jerarquía. Ejemplo los sistemas de archivos de una computadora.

Búsqueda: Almacenar información que queramos buscar rápidamente. Los árboles son más fáciles de buscar que una lista ligada. Algunos arboles tienen mejor rendimiento en la búsqueda (AVL y árboles rojo-negros).

Redes: los arboles son ideales para implementación en redes sociales o un juego de ajedrez.

Cómo hacer un árbol

```
Nodo<String> raiz = new Nodo<>("raiz");
```

Una vez que tenemos nuestra raíz, podemos agregar nuestro primer nodo secundario usando , que agrega un nodo secundario y lo asigna a un nodo primario. Nos referimos a este proceso como **inserción** (adición de nodos) y **eliminación** (eliminación de nodos).addChild.

```
Nodo<String> node1 = raiz.addChild(new Node<String>("nodo 1"));
```

Seguimos agregando nodos usando ese mismo proceso hasta que tengamos una estructura jerárquica compleja. En la siguiente sección, echemos un vistazo a los diferentes tipos de árboles que podemos usar.

Tipos de árboles

Existen una gran variedad de arboles que podemos usar para organizar los datos de manera diferente. El árbol que usamos depende del problema que estemos resolviendo.

Árbol binario completo

Existe un árbol binario completo cuando cada nivel, excluyendo el último, se llena y todos los nodos en el último nivel están tan a la izquierda como pueden ser.

árbol binario perfecto

Un árbol binario perfecto debe estar completo y completo. Todos los nodos interiores deben tener dos hijos, y todas las hojas deben tener la misma profundidad.

Arboles de búsqueda binaria

Un árbol de búsqueda binario es un árbol binario en el que cada nodo tiene una clave y un valor asociado. Esto permite la búsqueda rápida y ediciones (adiciones o eliminaciones), de ahí el nombre "búsqueda". Un árbol de búsqueda binario tiene condiciones estrictas basadas en su valor

Árbol AVL

Los árboles AVL son un tipo especial de árbol de búsqueda binaria que se auto equilibren comprobando el factor de equilibrio de cada nodo. El factor de equilibrio debe ser +1, 0o -1. La diferencia de altura máxima entre los subárboles izquierdo y derecho sólo puede ser una. Si esta diferencia se convierte en más de una, debemos reequilibrar nuestro árbol para que sea válido utilizando técnicas de rotación. Estos son más comunes para las aplicaciones donde la búsqueda es la operación más importante.

Árbol Rojo negro

Un árbol rojo negro tiene los nodos coloreados de rojo o negro, para ayudar a reequilibrar el árbol después de la inserción o eliminación. Ahorran tiempo con el equilibrio.

- La raíz siempre es negra
- Dos nodos rojos no pueden ser adyacentes (es decir, un padre rojo no puede tener un hijo rojo)
- Una ruta de acceso desde la raíz a una hoja debe contener el mismo número de nodos negros
- Un nodo nulo es negro