EDUCA2 – ¡Domina las Matemáticas y Programación!

Curso Intermedio de Programación en Python

Preparador: Alexanyer Naranjo

Guía Teórica N°5.1

Estructuras de Datos Avanzadas

En el Curso Básico conocimos sobre algunas estructuras de datos que Python define directamente como las Listas, Tuplas y Diccionarios. Pero ahora es turno de ir más allá y de conocer sobre estructuras de muy interesantes y más complejas que nos ayudarán a entender muy bien algunas bases de la teoría de la computación.

Cabe mencionar que existen una gran cantidad de estructuras de datos, las cuales listaremos al final de esta guía teórica e invitaremos a investigar mucho más sobre estas dado a que su utilidad dependerá del área donde queramos desenvolvernos.

Además de ello, en la siguiente guía teórica (**5.2**) hablaremos un poco sobre una estructura de datos interesante que se asemeja a los Árboles, pero no tiene el mismo comportamiento que estos, esta estructura de datos se conoce como **Grafo** y nos dedicaremos a esta únicamente en la próxima guía, mientras tanto, empecemos nuestro aprendizaje en este bloque del curso.

Conjuntos (Sets):

Muy parecidos a las Listas e inclusive a las Tuplas, los conjuntos nos permiten almacenar diferentes valores en una sola estructura, esto considerando que es una estructura No Homogénea, es decir, podemos almacenar valores de diferentes tipos de datos.

El único detalle diferente que tendrán los conjuntos en comparación a las Listas es que estos no almacenan elementos repetidos y esto es una ventaja dado a que en ocasiones no queremos redundar en la búsqueda de elementos pensando que nos podremos encontrar con más de uno igual, sino que podemos ser puntuales en la búsqueda del elemento que deseamos localizar dentro del conjunto.

Entre las operaciones de conjuntos, podemos encontrarnos con las siguientes operaciones básicas y métodos que nos ofrece Python para alterar el este del set:

- add: Nos permite agregar un nuevo elemento en nuestro conjunto.
- remove: Nos permite eliminar un elemento del conjunto.
- A **&** B: Realiza la intersección entre dos (2) conjuntos.
- A **|** B: Realiza la unión entre dos (2) conjuntos.
- A **–** B: Realiza la resta de conjuntos. Es importante saber que la resta no es conmutativa cuando a conjuntos estamos hablando.

Matrices:

Las matrices de manera muy sencilla son definidas como **Listas de Listas**, donde cada elemento perteneciente a la lista, será a su vez otra lista y podríamos generar matrices de 2, 3, 4, 5, …, N dimensiones según la cantidad de listas se encuentren indexadas.

Por facilidad de comprensión, podemos basarnos de una Matriz **NxM (N filas, M columnas)** o lo que también podríamos definir como una matriz de dos (2) dimensiones.

Las operaciones que podemos hacer con matrices en Python son similares a las Listas dado a que al final, cada elemento será a su vez una lista y podemos utilizar los métodos y funciones que Python nos ofrece para la manipulación de estas.

Estructuras FIFO - Pilas (Stacks):

Existen estructuras de datos que tienen un comportamiento en específico que son utilizadas en situaciones específicas que requieran llevar un control en especial de los datos almacenados.

Entre estas estructuras nos encontramos con las Pilas o de manera técnica, FIFO (First In – First Out), donde podemos imaginar una pila de libros en el cual, si queremos obtener un libro, tenemos que desapilar (**pop**) desde la cima cada uno de los libros hasta llegar al libro que deseamos. Y al momento de apilar (**push**), el último libro ingresado se encontrará en la cima de toda nuestra pila de libros.

Las pilas principalmente cuentan con solo dos operaciones fundamentales:

- pop: Utilizada para eliminar un elemento de la cima de la pila. Dicho elemento no tiene que regresar a la pila luego de haber sido desapilado.
- push: Utilizada para agregar un nuevo elemento a la cima de la pila.

Estructuras LIFO – Colas (Queues):

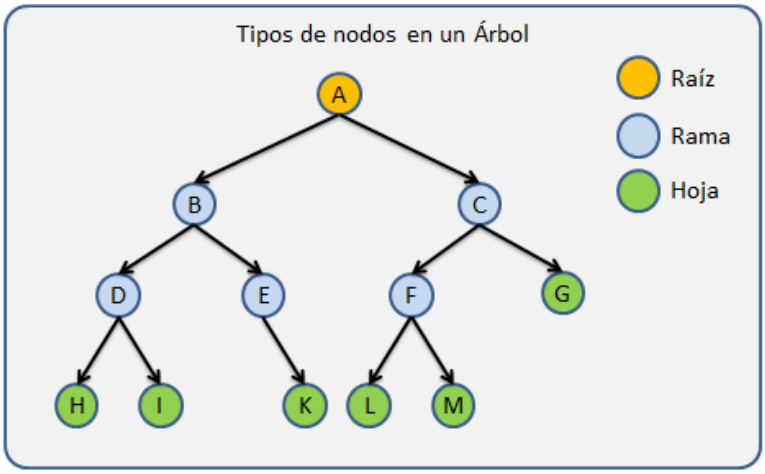
Otra estructura de datos que tiene un comportamiento en especial son las Colas, las cuales están definidas como LIFO (Last In – First Out), donde para entender su comportamiento podemos basarnos de una cola del supermercado. La primera persona en llegar, será la primera persona atendida en toda la cola.

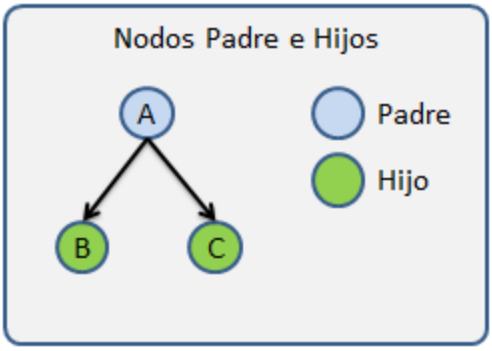
De igual forma que las pilas, las colas tienen solamente dos operaciones fundamentales, **add** (para eliminar un elemento) y **remove** (para agregar un nuevo elemento).

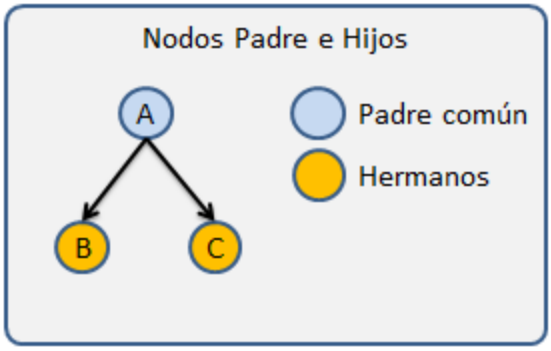
Árboles:

Los árboles son estructuras de datos complejas que poseen una gran variedad de implementaciones, pero para un primer vistazo y entender muy bien su funcionamiento, estaremos trabajando con árboles binarios, pero antes de ello, es importante conocer algunos términos teóricos que nos ayudarán a comprender estas estructuras tan interesantes.

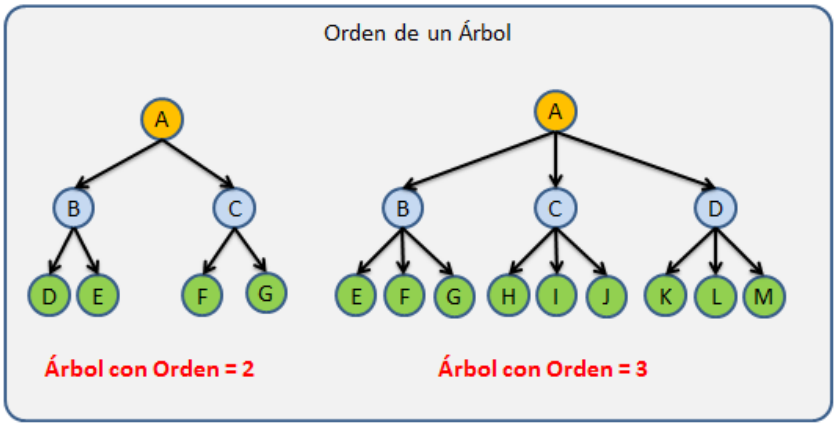
- **Nodos:** Se le llama Nodo a cada elemento que contiene un Árbol.
- **Nodo Raíz:** Se refiere al primer nodo de un Árbol, Solo un nodo del Árbol puede ser la Raíz.
- **Nodo Padre:** Se utiliza este término para llamar a todos aquellos nodos que tiene al menos un hijo.
- **Nodo Hijo:** Los hijos son todos aquellos nodos que tiene un padre.
- **Nodo Hermano:** Los nodos hermanos son aquellos nodos que comparte a un mismo padre en común dentro de la estructura.
- **Nodo Hoja:** Son todos aquellos nodos que no tienen hijos, los cuales siempre se encuentran en los extremos de la estructura.
- **Nodo Rama:** Estos son todos aquellos nodos que no son la raíz y que además tiene al menos un hijo.

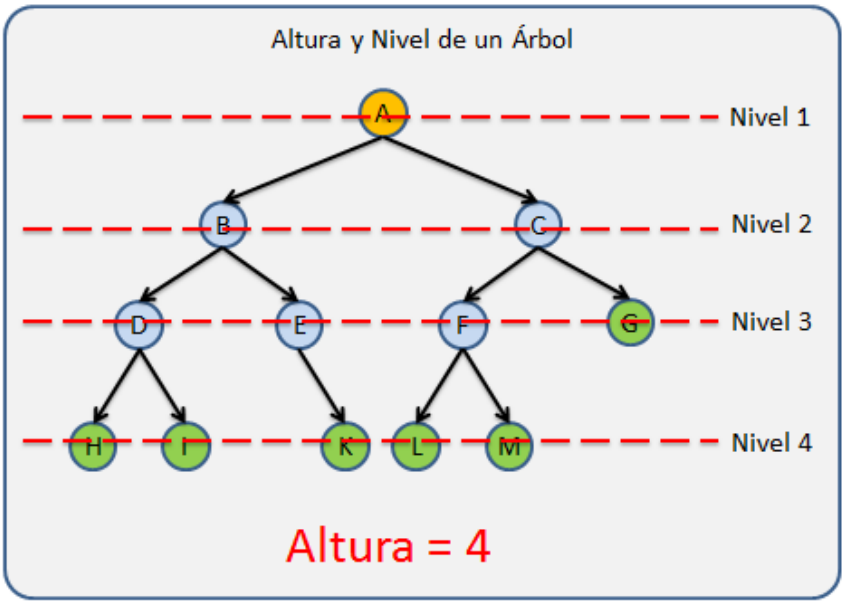






- **Orden:** El Orden de un árbol es el número máximo de hijos que puede tener un Nodo.
- **Nivel:** Nos referimos como nivel a cada generación dentro del árbol. Por ejemplo, cuando a un nodo hoja le agregamos un hijo, el nodo hoja pasa a ser un nodo rama, pero además el árbol crece una generación por lo que el Árbol tiene un nivel más. Cada generación tiene un número de Nivel distinto que las demás generaciones.
- **Altura:** Le llamamos Altura al número máximo de niveles de un Árbol.
- **Grado:** El grado se refiere al número mayor de hijos que tiene alguno de los nodos del Árbol y está limitado por el Orden, ya que este indica el número máximo de hijos que puede tener un nodo.





Árboles Binarios:

Esta estructura se caracteriza porque cada nodo solo puede tener máximo 2 hijos, dicho de otra manera, es un Árbol N - Ario de Grado 2 o también definido como un árbol N – Ario con N = 2.

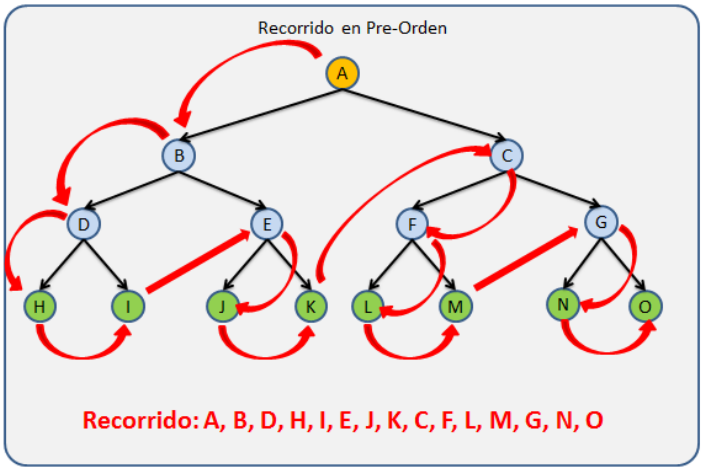
La importancia de los árboles, en especial los binarios, es que podemos realizar búsquedas eficientes de algún dato que se encuentre almacenado dentro de este dado a que no tendremos la tarea de buscar en el árbol completo, sino que podemos ir realizando cortes a la mitad dependiendo del valor que estemos tratando de localizar. De aquí, el motivo de que todos los elementos **menores** a la raíz se encuentren del lado izquierdo y todos los elementos **mayores** a la raíz se encuentren del lado derecho.

Recorrido de Árboles:

Existen diferentes tipos de recorridos para atravesar cada uno de los elementos pertenecientes a un árbol, hablaremos de los tres (3) más importantes; PreOrden, InOrden y PostOrden.

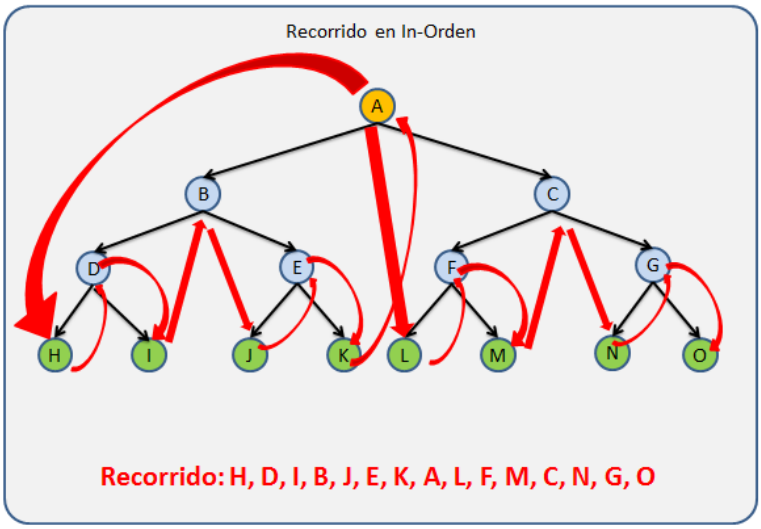
Recorrido PreOrden:

El recorrido inicia en la Raíz y luego se recorre en PreOrden cada uno de los sub-árboles de izquierda a derecha **(Raíz -> Izquierda -> Derecha)**.



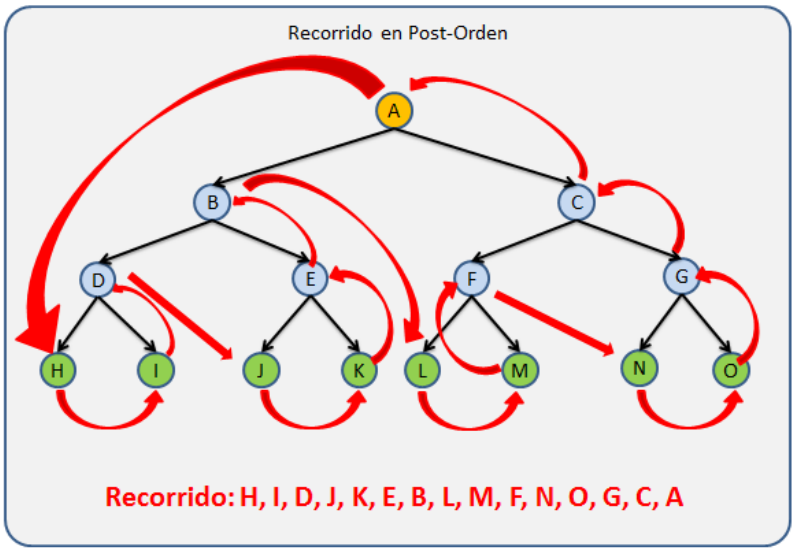
Recorrido InOrden:

Se recorre en InOrden el primer sub-árbol, luego se recorre la raíz y al final se recorre en InOrden los demás sub-árboles **(Izquierda -> Raíz -> Derecha).**



Recorrido PostOrden:

Se recorre el pos-orden cada uno de los sub-árboles y al final se recorre la raíz. **(Izquierda -> Derecha -> Raíz).**



Algunas otras estructuras de datos:

A continuación, se listan el nombre de otras estructuras de datos avanzadas muy interesantes las cuales invitamos a todos a investigar sobre estas:

- Colas de Prioridades
- Árboles AVL
- Árboles de toma de decisión
- Árboles Rojos y Negros
- Tablas Hash
- Mapas
- Árboles B – Tree
- Árboles Trie
- Grafos

¡Nos vemos en la próxima clase!

Aún no hemos finalizado nuestra travesía a través de las estructuras de datos avanzadas. En la próxima guía hablaremos de una estructura muy interesante la cuales son los grafos. Así que no nos detengamos y continuemos con nuestro aprendizaje con Python.