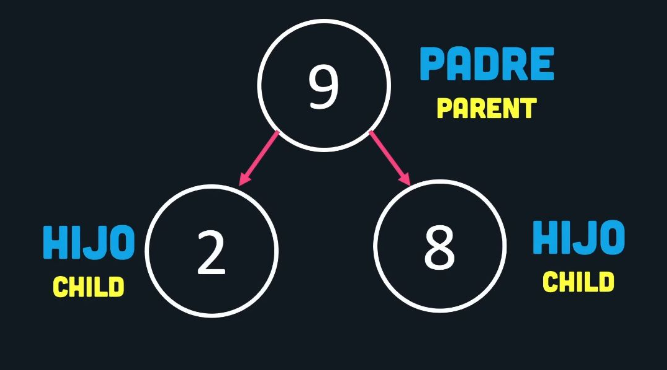


Los ***TREE*** son una estructura de datos recursiva.



Forma

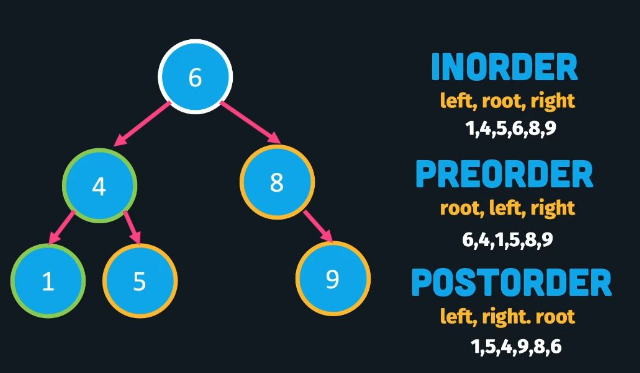
Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

Diagrama, Esquemático

Descripción generada automáticamente con confianza media



Con el siguiente código cumplimos con la estructura básica del árbol:

class TreeNode:

    def \_\_init\_\_(self, data):

        self.data = data

        self.children = []

        self.parent = None

Código para agregar el Child y Parent.

class TreeNode:

    def \_\_init\_\_(self, data):

        self.data = data

        self.children = []

        self.parent = None

    def add\_child(self, child):

        child.parent = self

        self.children.append(child)

def build\_product\_tree():

    root = TreeNode("Eletronics")

    laptop = TreeNode("Laptop")

    root.add\_child(laptop)

build\_product\_tree()

NOTA: en el siguiente código, SELF, es ROOT y CHILD es LAPTOP

    def add\_child(self, child):

        child.parent = self

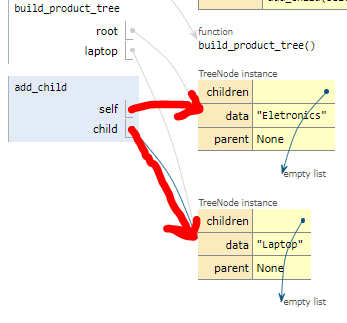
        self.children.append(child)

def build\_product\_tree():

    root = TreeNode("Eletronics")

    laptop = TreeNode("Laptop")

    root.add\_child(laptop)



En la segunda línea de la función ADD\_CHILDREN, el parent se agrega al laptop. Es decir, ROOT que es SELF, se agregara al PARENT de LAPTOP.

    def add\_child(self, child):

        child.parent = self

        self.children.append(child)

Diagrama

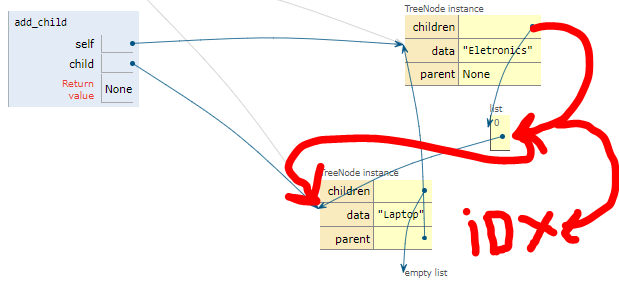
Descripción generada automáticamente

En la tercera linea, se esta agregando LAPTOP al SELF.CHILDREN que es una lista, de SELF o ROOT

    def add\_child(self, child):

        child.parent = self

        self.children.append(child)



Creando un TREE de eletrodomesticos

def build\_product\_tree():

    root = TreeNode("Eletronics")

    laptop = TreeNode("Laptop")

    laptop.add\_child(TreeNode("Mac"))

    laptop.add\_child(TreeNode("Surface"))

    laptop.add\_child(TreeNode("Thinkpad"))

    cellphone = TreeNode("cell Phone")

    cellphone.add\_child(TreeNode("Iphone"))

    cellphone.add\_child(TreeNode("Samsung"))

    cellphone.add\_child(TreeNode("Huawei"))

    tv = TreeNode("TV")

    tv.add\_child(TreeNode("Samsung"))

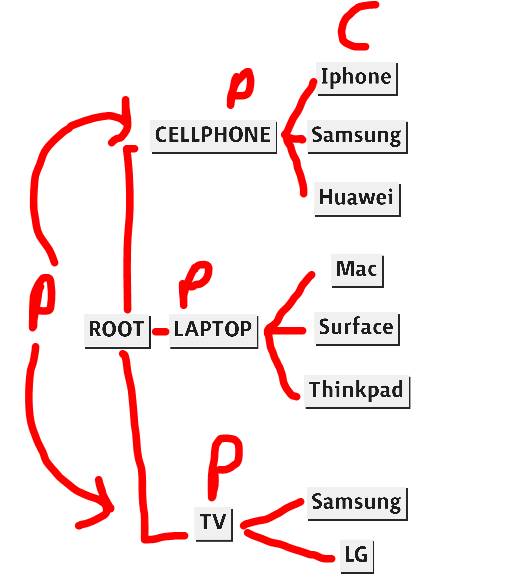
    tv.add\_child(TreeNode("LG"))

    root.add\_child(laptop)

    root.add\_child(cellphone)

    root.add\_child(tv)

build\_product\_tree()



Si imprimiros los hijos dentro del ***ROOT*** no saldría:

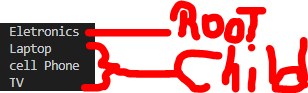
    def print\_tree(self):

        print(self.data)

        for child in self.children:

            print(child.data)

    root.print\_tree()



***laptop.print\_tree()***

Texto

Descripción generada automáticamente

***cellphone.print\_tree()***

Texto

Descripción generada automáticamente

***tv.print\_tree()***

Texto

Descripción generada automáticamente

Como dato curioso si se puede acceder a los hojas a través del ROOT

Texto

Descripción generada automáticamente con confianza media

***RECURSION:***

    def print\_tree(self):

        print(self.data)

        if self.children:

            for child in self.children:

                child.print\_tree()

    root.print\_tree()

***Orden de ejecución de recursividad:***

primero itera en el primer nodo, que es ROOT > luego como el Loop FOR esta iterando entre los hijos, ITERA entre LAPTOP > LAPTOP llama recursivamente a la FUNCION > IMPRIME todos los NODO.DATA dentro de LAPTOP > y cuando no haya más, EL FOOR VUELVE al ROOT y continua con el SIGUIENTE NODO que es CELLPHONE a hacer lo mismo que con LAPTOP.

***OBTENER EL NIVEL:***

Para obtener el nivel debemos contar los padres.

    def get\_level(self):

        level = 0

        p = self.parent

        while p:

            level += 1

            p = p.parent

        return level

    print(root.get\_level())

RESULTADO == 0

    print(laptop.get\_level())

RESULTADO = 1

Agregar prefijos:

    def print\_tree(self):

        spaces = ' ' \* self.get\_level() \* 3

        prefix = spaces + "|\_\_" if self.parent else ""

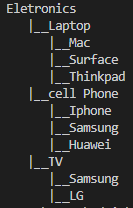
        print(prefix + self.data)

        if self.children:

            for child in self.children:

                child.print\_tree()

EL PRIMER IF NO SE EJECUTA, PORQUE EL NODO ROOT, NO TIENE UN PADRE



RODDY (CEO)

|\_\_ YOSI (CTO)

|\_\_ JENNY (INFRASTRUCTURE HEAD)

|\_\_ ANA (CLOUD MANAGER)

|\_\_ VERCELLI (APP MANAGER)

|\_\_ EDDIE (APLICATION HEAD)

|\_\_ RAFAEL (HR HEAD)

|\_\_ ROSARIO (RECRUITMENT MANAGER)

|\_\_ TEJEDA (POLICY MANAGER)