



## Correcciones realizadas al árbol AVL

El árbol presentaba problemas con las rotaciones, “rotate\_right()” y “rotate\_left()” no asignaban el resultado ya que no retornaba valores, modificamos el programa agregando rotaciones dobles para cada sentido, rotaciones doble a la derecha y rotaciones doble a la izquierda manteniendo correctamente el balance AVL después de cada inserción y agregamos “return” a las funciones que les hacía falta.

## Funciones agregadas al árbol AVL

- **Función In-order:** Esta función permite el ordenamiento correcto del árbol, valida el BST del árbol y permite verificar después de cada operación
- **Visualización del árbol:** Permite imprimir o mostrar una estructura jerárquica del ordenamiento del árbol, además de que incluye la altura y el balance de cada nodo y muestra una representación de la relación padre-hijo.

## Casos de prueba:

### Caso 1:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
EDD\Solución\Lab_AVL-corregido.py
Insertando valores: [210, 200, 305, 410, 750, 425]

--- Después de inserciones ---
^- 410
|^- 210
| |^- 200
| |^- 305
|^- 750
|^- 425
^- 410
|^- 210
| |^- 200
| |^- 305
|^- 750
|^- 425

--- Detalles de los nodos ---
(Nodo = 200: Altura = 1, Balance = 0)
(Nodo = 305: Altura = 1, Balance = 0)
(Nodo = 210: Altura = 2, Balance = 0)
(Nodo = 425: Altura = 1, Balance = 0)
(Nodo = 750: Altura = 2, Balance = 1)
(Nodo = 410: Altura = 3, Balance = 0)

--- Recorrido In-orden ---
[200, 210, 305, 410, 425, 750]
PS D:\Valler EDD\Solución>
```

### Caso 2:

```
Insertando valores: [2, 540, 35, 70, 1081, 54]

--- Después de inserciones ---
^- 70
|^- 35
| |^- 2
| |^- 54
|^- 540
|^- 1081
^- 70
|^- 35
| |^- 2
| |^- 54
|^- 540
|^- 1081

--- Detalles de los nodos ---
(Nodo = 2: Altura = 1, Balance = 0)
(Nodo = 54: Altura = 1, Balance = 0)
(Nodo = 35: Altura = 2, Balance = 0)
(Nodo = 1081: Altura = 1, Balance = 0)
(Nodo = 540: Altura = 2, Balance = -1)
(Nodo = 70: Altura = 3, Balance = 0)

--- Recorrido In-orden ---
[2, 35, 54, 70, 540, 1081]
PS D:\Valler EDD\Solución>
```

### Caso 3:

```
Insertando valores: [14, 40, 350, 790, 2540, 3150]

--- Después de inserciones ---
'- 790
  |- 40
  | |- 14
  | |- 350
  |- 2540
  '- 3150
'- 790
  |- 40
  | |- 14
  | |- 350
  |- 2540
  '- 3150

--- Detalles de los nodos ---
(Nodo = 14: Altura = 1, Balance = 0)
(Nodo = 350: Altura = 1, Balance = 0)
(Nodo = 40: Altura = 2, Balance = 0)
(Nodo = 3150: Altura = 1, Balance = 0)
(Nodo = 2540: Altura = 2, Balance = -1)
(Nodo = 790: Altura = 3, Balance = 0)

--- Recorrido In-orden ---
[14, 40, 350, 790, 2540, 3150]
PS D:\Taller EDO\Solución> []
```

Al introducir cada una de estas correcciones y mejora ha permitido transformar el código con una implementación más confiable y manteniendo la propiedad AVL verificando automáticamente el orden de los datos, además de imprimir detalladamente como se ordenaron los datos.