HANDSON-10

DESIGN ANALYSIS AND ALGORITHMS

RASHMITHA RAMASANI

ID:1002233393

BINARY SEARCH TREE:

Overview

This repository contains a Python implementation of a Binary Search Tree (BST) with basic operations like adding, finding, and removing nodes, as well as in-order traversal.

Features

- **Add Nodes**: Insert keys while maintaining BST properties.
- **Find Nodes**: Search for keys in the tree.
- **Remove Nodes**: Delete keys and adjust the tree.
- **In-Order Traversal**: Retrieve keys in sorted order.

Classes

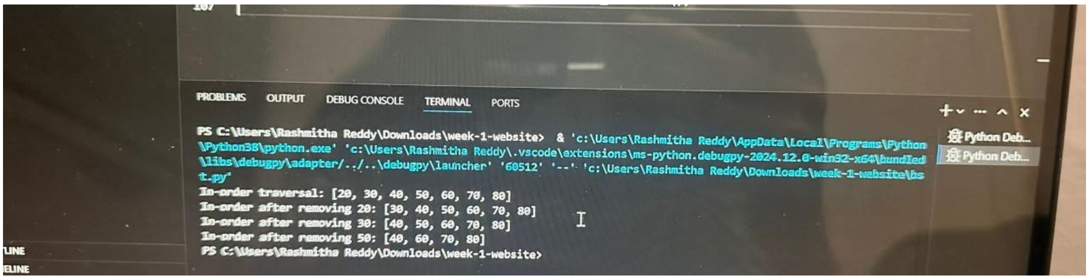
Node

Represents a single node in the BST.

BinarySearchTree

Manages the tree with methods:

- add(key): Insert a key.
- find(key): Search for a key.
- remove(key): Delete a key.
- inorder_traversal(): Get sorted keys.
- OUTPUT:



```
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>  & 'c:\Users\Rashmitha Reddy\AppData\Local\Programs\Python
\Python38\python.exe' 'c:\Users\Rashmitha Reddy\.vscode\extensions\ms-python.debugpy-2024.12.0-win32-x64\bundled
\libs\debugpy\adapter/../..\debugpy\launcher' '60512' '--' 'c:\Users\Rashmitha Reddy\Downloads\week-1-website\bs
t.py'
In-order traversal: [20, 30, 40, 50, 60, 70, 80]
In-order after removing 20: [30, 40, 50, 60, 70, 80]
In-order after removing 30: [40, 50, 60, 70, 80]
In-order after removing 50: [40, 60, 70, 80]
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>
```

-

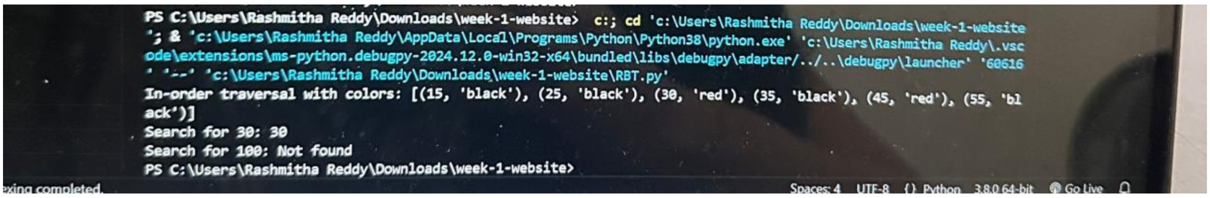
RBT:

Overview

- This repository contains a Python implementation of a Red-Black Tree, a type of self-balancing binary search tree. This implementation supports insertion, searching, and in-order traversal of the tree, maintaining the Red-Black properties.

- **Features**
- **Insert Nodes**: Add values while preserving tree balance.
- **Search Nodes**: Find values in the tree.
- **In-Order Traversal**: Retrieve values in sorted order along with their colors.
- **Classes**
- **RBTNode**
- Represents a node in the Red-Black Tree with attributes:
- value: The key stored in the node.
- color: The color of the node (either 'red' or 'black').
- left: Pointer to the left child.
- right: Pointer to the right child.
- parent: Pointer to the parent node.
- **RedBlackTree**
- Manages the tree with methods:
- insert(value): Insert a value into the tree.
- search(value): Search for a value and return the corresponding node.
- inorder_traversal(): Get a list of values and their colors in sorted order.

output



```
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>  c:; cd 'c:\Users\Rashmitha Reddy\Downloads\week-1-website
'; & 'c:\Users\Rashmitha Reddy\AppData\Local\Programs\Python\Python38\python.exe' 'c:\Users\Rashmitha Reddy\.vsc
ode\extensions\ms-python.debugpy-2024.12.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '60616
' '--' 'c:\Users\Rashmitha Reddy\Downloads\week-1-website\RBT.py'
In-order traversal with colors: [(15, 'black'), (25, 'black'), (30, 'red'), (35, 'black'), (45, 'red'), (55, 'bl
ack')]
Search for 30: 30
Search for 100: Not found
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>
```

AVL:

Overview

This repository contains a Python implementation of an AVL Tree, a type of self-balancing binary search tree. The AVL Tree maintains its balance through rotations during insertion and deletion operations, ensuring that the tree remains approximately balanced.

Features

- **Insert Nodes**: Add values while maintaining tree balance.
- **Delete Nodes**: Remove values and adjust the tree to maintain balance.
- **Search Nodes**: Find values efficiently.
- **In-Order Traversal**: Retrieve values in sorted order.

Classes

AVLTreeNode

Represents a node in the AVL Tree with attributes:

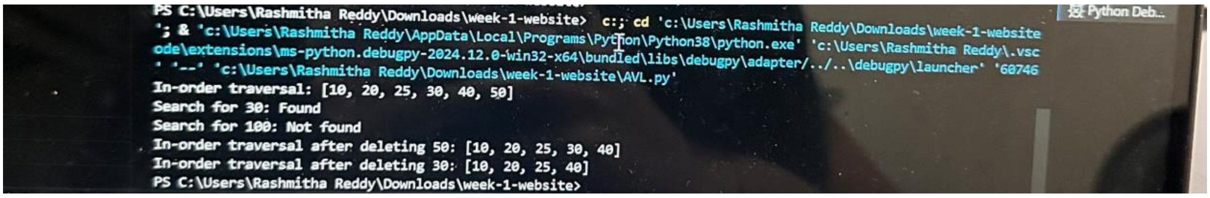
- value: The key stored in the node.
- left_child: Pointer to the left child.
- right_child: Pointer to the right child.
- height: The height of the node.

AVLTree

Manages the AVL Tree with methods:

- insert(value): Insert a value into the tree.
- delete(value): Remove a value from the tree.
- search(value): Search for a value and return the corresponding node.
- inorder_traversal(): Get a list of values in sorted order.

Output:



```
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>  c:; cd 'c:\Users\Rashmitha Reddy\Downloads\week-1-website
'; & 'c:\Users\Rashmitha Reddy\AppData\Local\Programs\Python\Python38\python.exe' 'c:\Users\Rashmitha Reddy\.vsc
ode\extensions\ms-python.debugpy-2024.12.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\launcher' '60746
' '--' 'c:\Users\Rashmitha Reddy\Downloads\week-1-website\AVL.py'
In-order traversal: [10, 20, 25, 30, 40, 50]
Search for 30: Found
Search for 100: Not found
In-order traversal after deleting 50: [10, 20, 25, 30, 40]
In-order traversal after deleting 30: [10, 20, 25, 40]
PS C:\Users\Rashmitha Reddy\Downloads\week-1-website>
```