AVLTree Report

Emirhan Altunel

Implementation

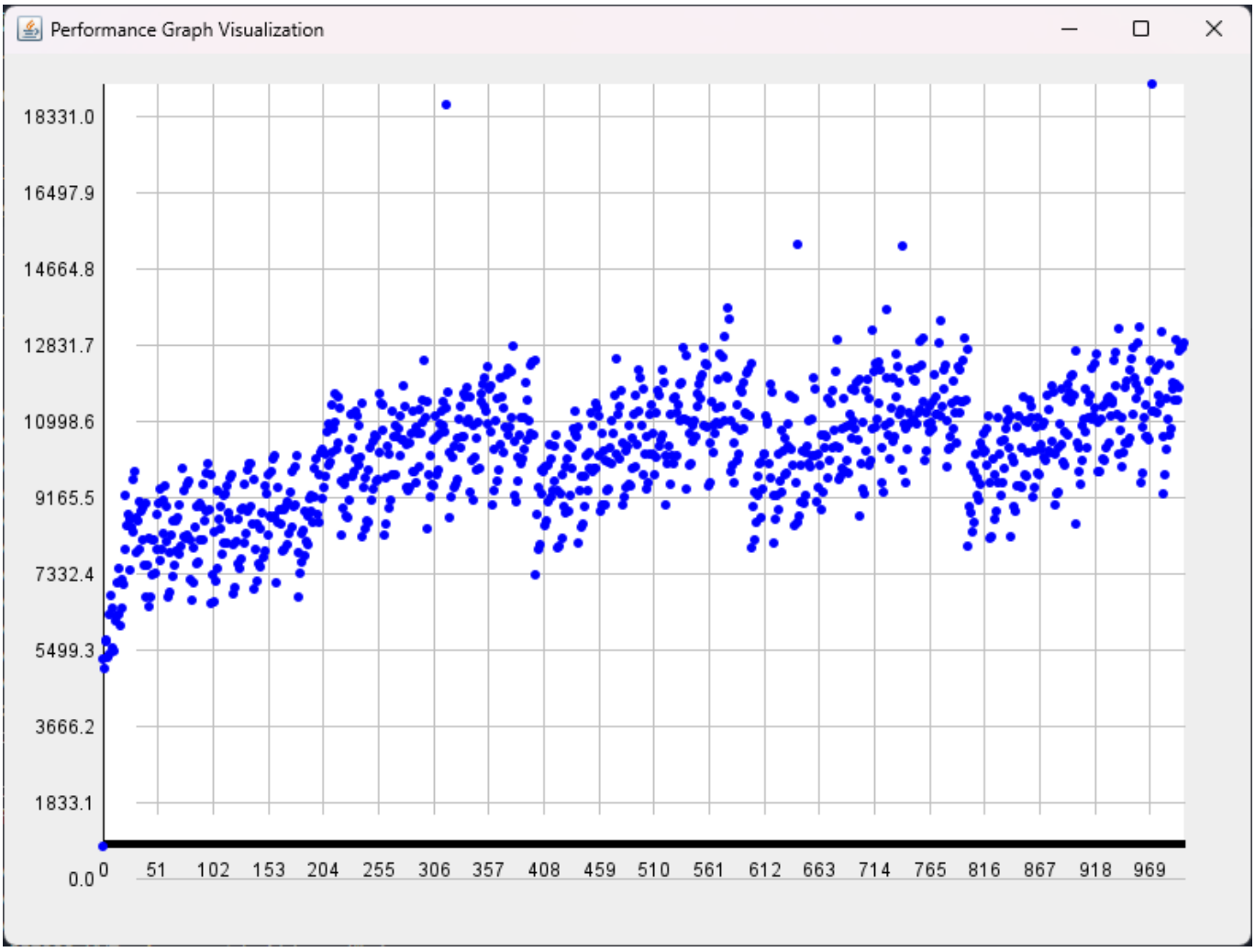
I implemented the AVLTree with the following methods:

- `add` : Adds a new node to the tree.
- `balance` : Balances the tree.
- `remove` : Removes a node from the tree.
- `update` : Updates a node in the tree.
- `search` : Searches for a node in the tree.

Add

The `add` method adds a new node to the tree. It first adds the node to the tree like a normal binary search tree. Then it runs the `balance` method to balance the tree.

Test Results



Balance

The `balance` method balances the tree. It first calculates the balance factor of the node (the difference between the left and right subtree heights). If the balance factor is greater than 1, it rotates the tree to the right. If the balance factor is less than -1, it rotates the tree to the left.

Right Rotation

The right rotation is done when the balance factor is greater than 1. It rotates the tree to the right. The left child of the node becomes the new root of the subtree.

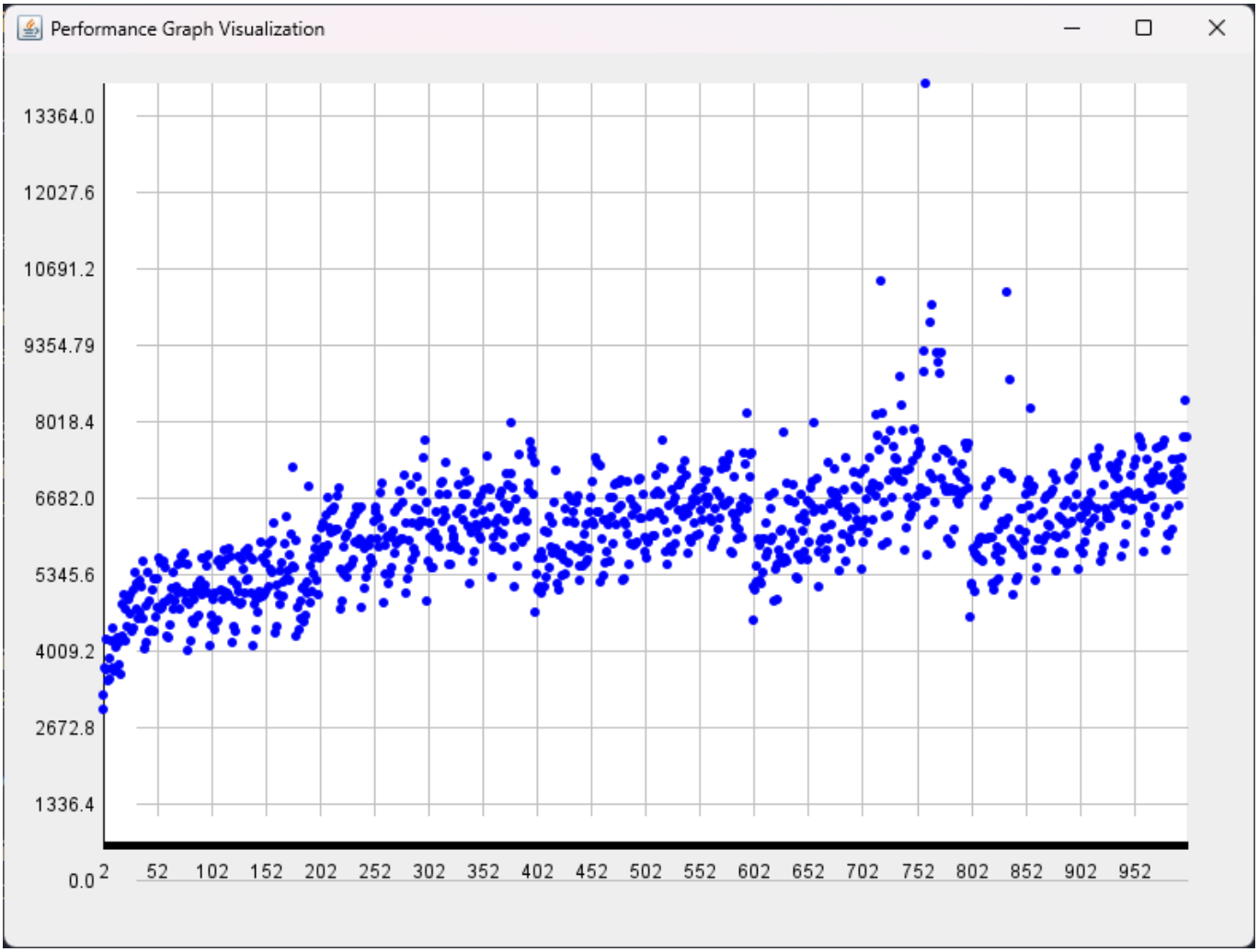
Left Rotation

The left rotation is done when the balance factor is less than -1. It rotates the tree to the left. The right child of the node becomes the new root of the subtree.

Remove

The `remove` method removes a node from the tree. It first removes the node like a normal binary search tree. Then it runs the `balance` method to balance the tree.

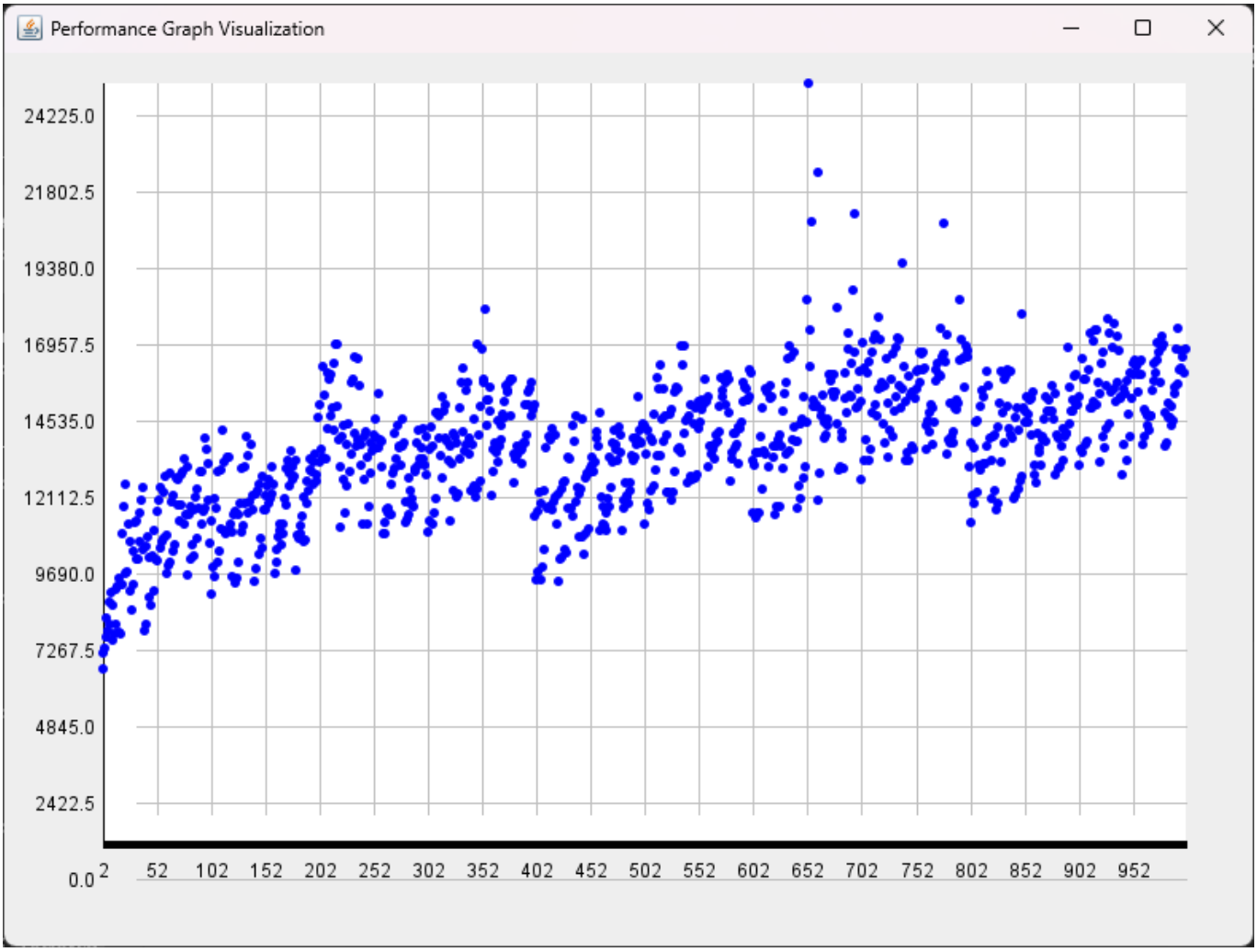
Test Results



Update

The `update` method updates a node in the tree. If the node name will be updated, it first removes the node from the tree. Then it adds the new node to the tree for preserving the AVLTree property.

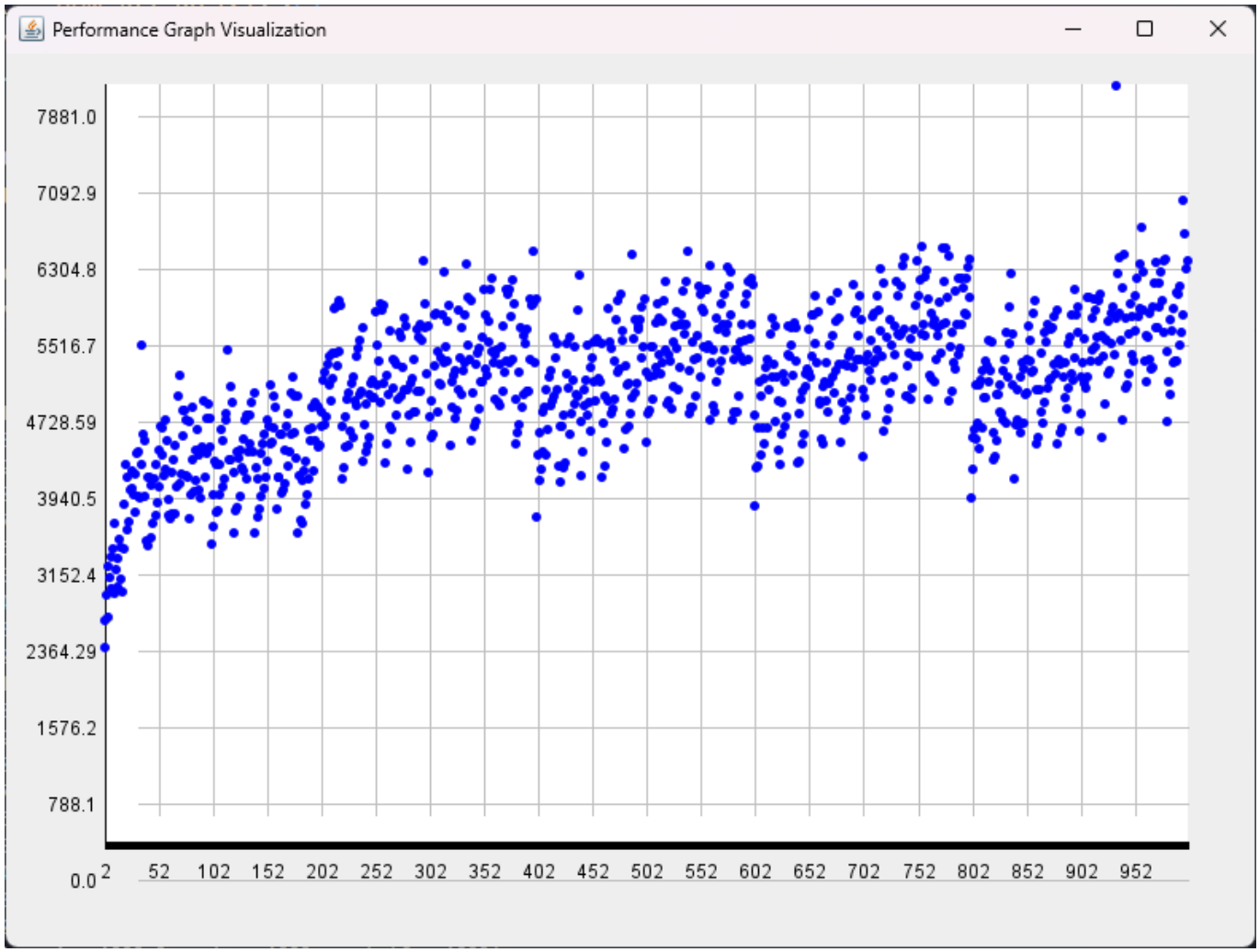
Test Results



Search

The `search` method searches for a node in the tree. It uses the binary search tree search algorithm to find the node.

Test Results


Performance Graph Visualization
7881.0
7092.9
6304.8
5516.7
4728.59
3940.5
3152.4
2364.29
1576.2
788.1
0.0
2
52
102
152
202
252
302
352
402
452
502
552
602
652
702
752
802
852
902
952