Documentation for Red-Black Tree Operations

Victor Racelescu

Contents

1 Introduction

This project is an interactive program to perform operations on a Red-Black Tree, including insertion, deletion, traversal, and visualization. The visualization is implemented using the SFML library to enhance the user experience.

2 How to Use the Program

The program is menu-driven, and users can interact by entering integers corresponding to the menu options. The tree nodes can only have integer keys, and if a key already exists in the tree, it will not be added again. Some options include a visual representation of the tree to make it easier to understand the structure and the result of specific operations.

2.1 Menu Options

Below is the list of available menu options, along with their descriptions:

1. **Add nodes**: Enter one or more integers separated by spaces to insert them into the Red-Black Tree. Duplicate keys are not allowed.
2. **Delete node**: Enter the key of the node to be deleted from the tree. The tree will be rebalanced according to the Red-Black Tree properties.
3. **Get minimum node**: Displays the minimum key in the tree. The minimum node will be highlighted in **yellow** in the visualization.
4. **Get maximum node**: Displays the maximum key in the tree. The maximum node will be highlighted in **yellow** in the visualization.
5. **Get successor of a node**: Enter the key of a node to find its successor. If it exists, the successor node will be highlighted in **yellow** in the visualization.
6. **Get predecessor of a node**: Enter the key of a node to find its predecessor. If it exists, the predecessor node will be highlighted in **yellow** in the visualization.
7. **Show tree**: Displays the entire Red-Black Tree structure as a graphical visualization.
8. **Show inorder traversal**: Displays the inorder traversal of the tree, which prints the keys of the tree nodes in ascending order.
9. **Show black-height of the tree**: Prints the black-height of the tree, which is the number of black nodes on any path from the root to a leaf.
10. **Show the maximum key of a black node in the tree**: Displays the maximum key among all black nodes in the tree. The node will be highlighted in **yellow** in the visualization.
11. **Show the maximum key of a red node in the tree**: Displays the maximum key among all red nodes in the tree. The node will be highlighted in **yellow** in the visualization.
12. **Show tree depth**: Prints the depth of the tree, which is the maximum number of edges from the root to any leaf node.
13. **Exit**: Exits the program.

3 Visualizations

The visualization uses the SFML library to display the structure of the tree and highlight specific nodes during the following operations:

- **Get minimum node (Option 3)**: Highlights the minimum node.
- **Get maximum node (Option 4)**: Highlights the maximum node.
- **Get successor of a node (Option 5)**: Highlights the successor node.
- **Get predecessor of a node (Option 6)**: Highlights the predecessor node.
- **Show tree (Option 7)**: Displays the full tree structure.
- **Show the maximum key of a black node (Option 10)**: Highlights the black node with the maximum key.
- **Show the maximum key of a red node (Option 11)**: Highlights the red node with the maximum key.

4 Conclusion

This project provides a comprehensive set of operations for Red-Black Tree manipulation and visualization, making it a useful tool for learning and understanding Red-Black Trees.