Consider an AVL Tree implementation in C with the following values to be inserted and deleted. Use the provided values to demonstrate the correctness of your AVL Tree implementation. Assume that each node in the AVL Tree contains an integer key and a balance factor.

Values to be Inserted: 50, 30, 70, 20, 40, 60, 80

Values to be Deleted: After insertion, perform the following deletions:

- Delete 30
- Delete 80
- Delete 40

Implement the AVL Tree operations based on the provided values and perform the following tasks:

1. **Insertion (20 marks):**
    - Implement the insertion operation to build the AVL Tree with the given values.
2. **Deletion (30 marks):**
    - Implement the deletion operation to remove the specified values and trigger the necessary rotations.
    - Provide the AVL Tree structure after each deletion operation, illustrating the changes made to maintain the AVL property through Inorder, Preorder and Postorder traversals.

Note:

- ✓ Marks are allocated based on correctness, efficiency, and clarity of the code. Make sure to adhere to good programming practices.
- ✓ The provided values are selected to test the rotation operations and the ability to maintain the AVL property during deletions.