Task 1: Write Algo for AVL tree

Solution 1 : Algorithm / Steps for AVL Tree (Self-Balancing Binary Search Tree)

1. Start with an empty tree.

2. Insert a new node as in a regular Binary Search Tree (BST).

3. After insertion, check the balance factor (height difference) of each node from the inserted node up to the root.

4. If the balance factor of any node becomes greater than 1 or less than -1, the tree is unbalanced.

5. Perform one of the following rotations to balance the tree:

a. Left Rotation

b. Right Rotation

c. Left-Right Rotation

d. Right-Left Rotation

6. Repeat steps 2-5 for every insertion or deletion.

7. After every operation, ensure the tree remains balanced (balance factor of every node is -1, 0, or 1).

8. End.

Solution 2 : AVL search Algo

1 − Create a node

2 − Check if tree is empty

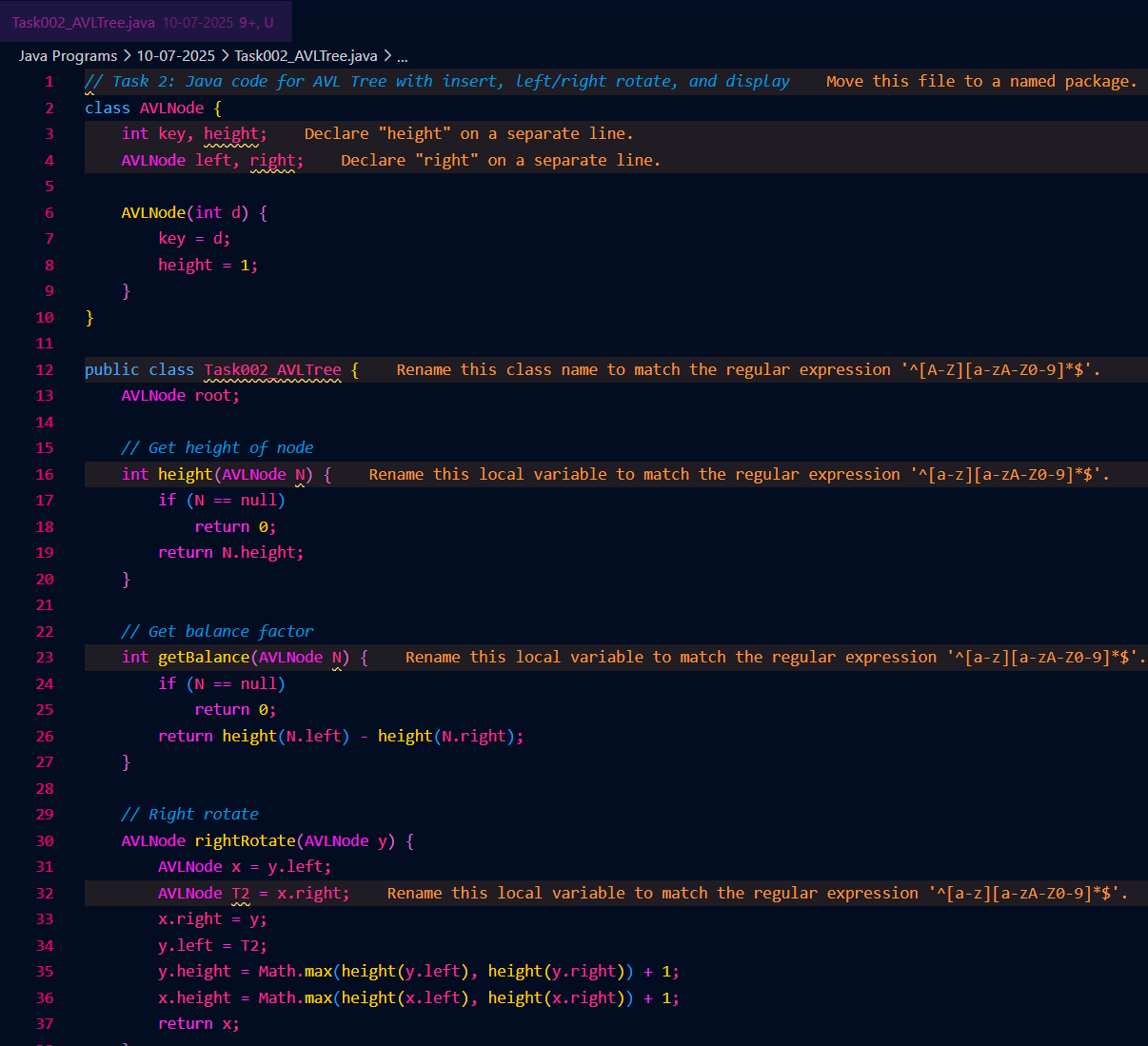
3 − If tree is empty, new node is root node.

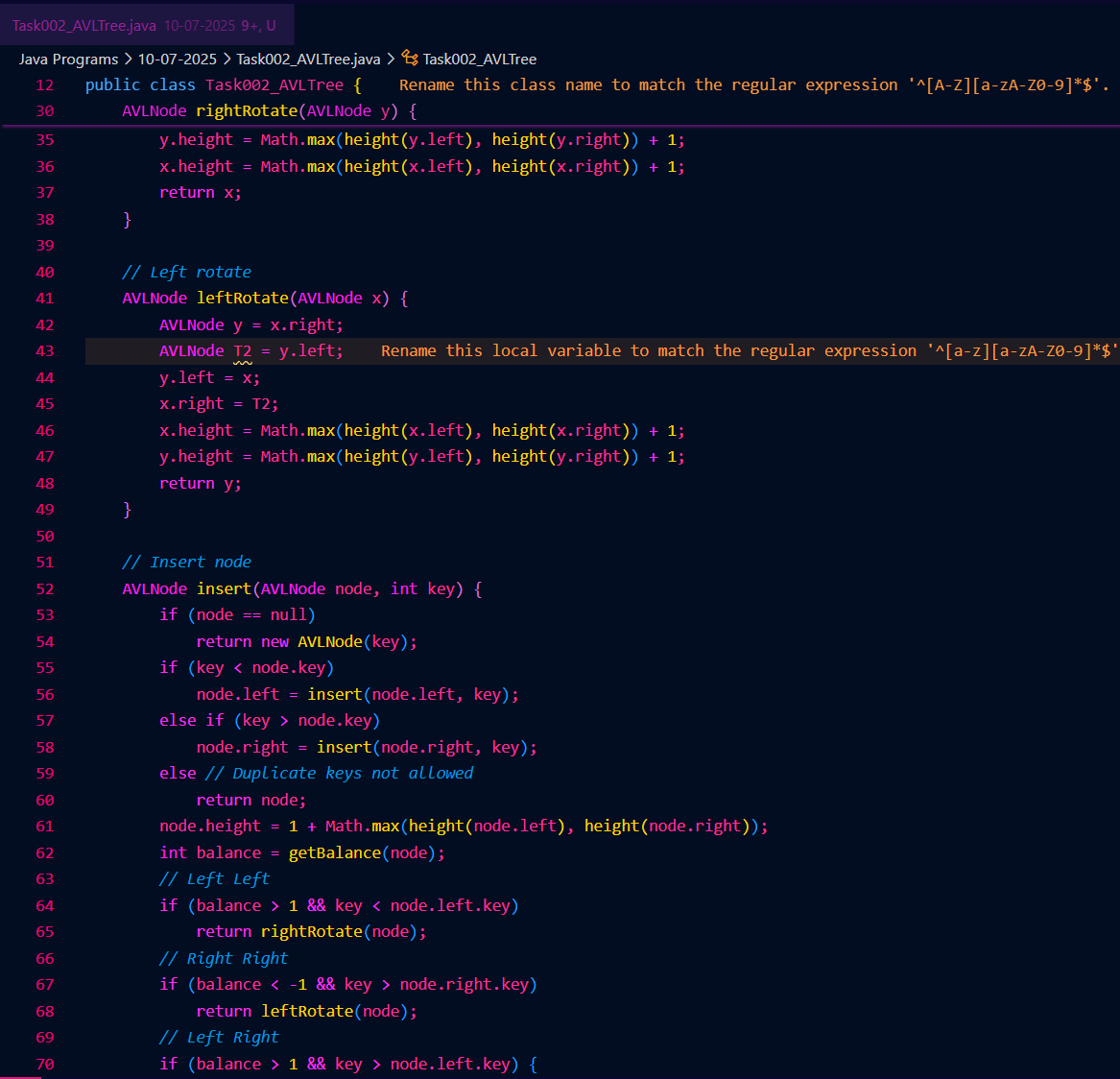
4 − not empty, perform Binary Search Tree insertion operation and check balancing factor of the node in the tree.

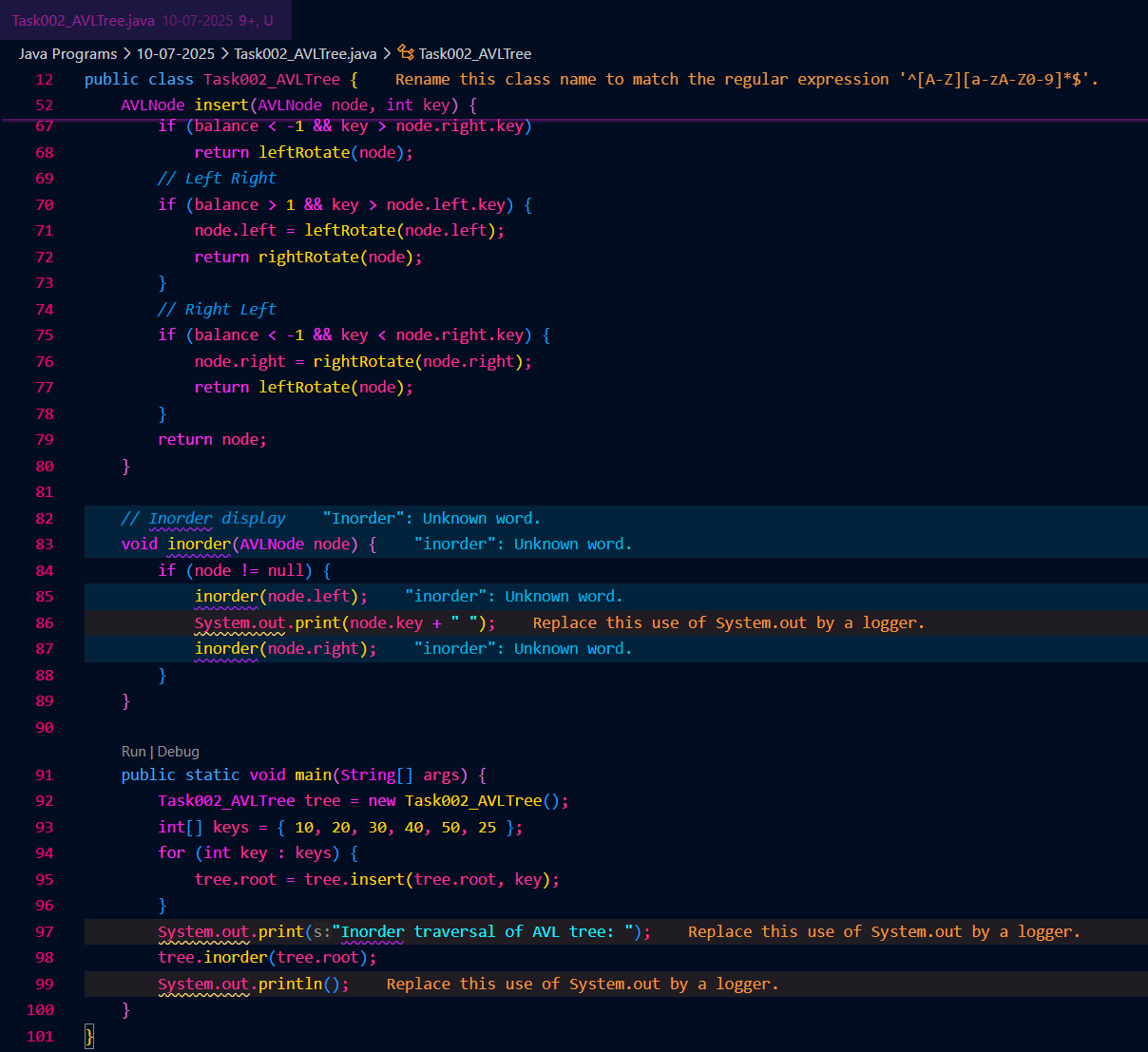
5 − Suppose balancing factor > apply rotations on node and resume insertion from Step 4.

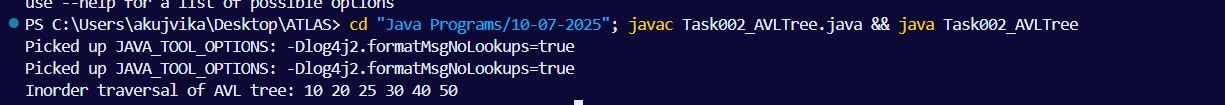
Task 2: Write code for AVL tree (Hint: try to insert nodes , While inserting get the balance of the tree , Create 2 methods for left rotate and right rotate , Try to insert , Finally display)

Solution :









Task 3: Write algo for Read Black tree insertion

Solution 1 : Algorithm / Steps for Red-Black Tree Insertion :

1. Insert the new node as in a regular Binary Search Tree (BST), color it RED.

2. If the new node is the root, color it BLACK and stop.

3. If the parent of the new node is BLACK, stop (tree is still valid).

4. If the parent is RED:

a. If the uncle is RED:

- Color the parent and uncle BLACK.

- Color the grandparent RED.

- Move up to the grandparent and repeat from step 2.

b. If the uncle is BLACK or null:

- If the new node is on the "inside" (left-right or right-left), rotate to make it "outside" (left-left or right-right).

- Perform a rotation on the grandparent (left or right, as needed).

- Swap colors of parent and grandparent.

5. Ensure the root is always BLACK.

6. End.

Solution 2 : Insert an Element - Red Black Tree −

1. Check tree is empty. If empty, then insert new node - color Black. (Because Root Node - Black in color)

2. else if Tree - not empty then insert new node as leaf node to the end and color - Red.

3. If parent of new node is Red and its neighbours(parent’s) node is also Red,

then Flip the color of the both neighbour and Parent and Grandparents (If it is not Root Node Otherwise Flip the color of the Parent and neighbour only) i.e., Black.

4. If parent of new node is Red and its neighbours(parent’s) node is empty or NULL,

then Rotate (either Left-Left or Left-Right rotation) the new node and parent.

5. we have two types of rotation

- Left Left Rotation and

- Left Right Rotation.

6. we apply Rotation in some conditions only.

The conditions are −

- If parent of new node is Red and neighbour node is empty or NULL, then rotate left or right rotation.

- In Left-Left Rotation flip the color of the parent and grandparent.

Make the parent as Grandparent and grandparent as child.

Task 4: Wap to insert an element in red black tree

Solution :

