LAB_07

ICS 202

Jalal Ali Zainaddin | 202154790 | 10/25/2023

Task 1: InsertAVL(), rotateRight(),rotateRightLeft ():

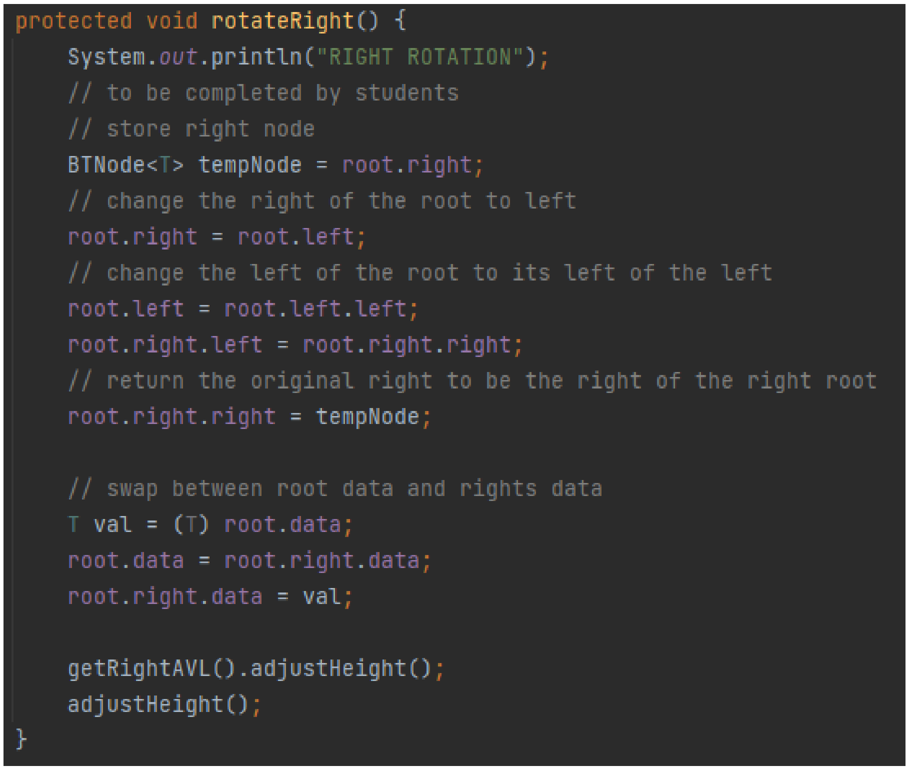
rotateRight():

First we store right node, then change the right of the root to left, then change the left of the root to its left of the left, and return the original right to be the right of the right root, Finally, we swap between root data and the rights data.

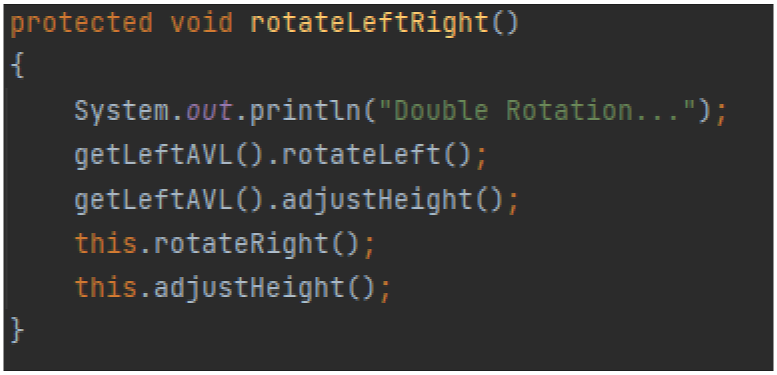
rotateRightLeft():

Take the right part of the tree and do right rotation, adjust the height, then do left rotation by using rotateLeft() method

Code:

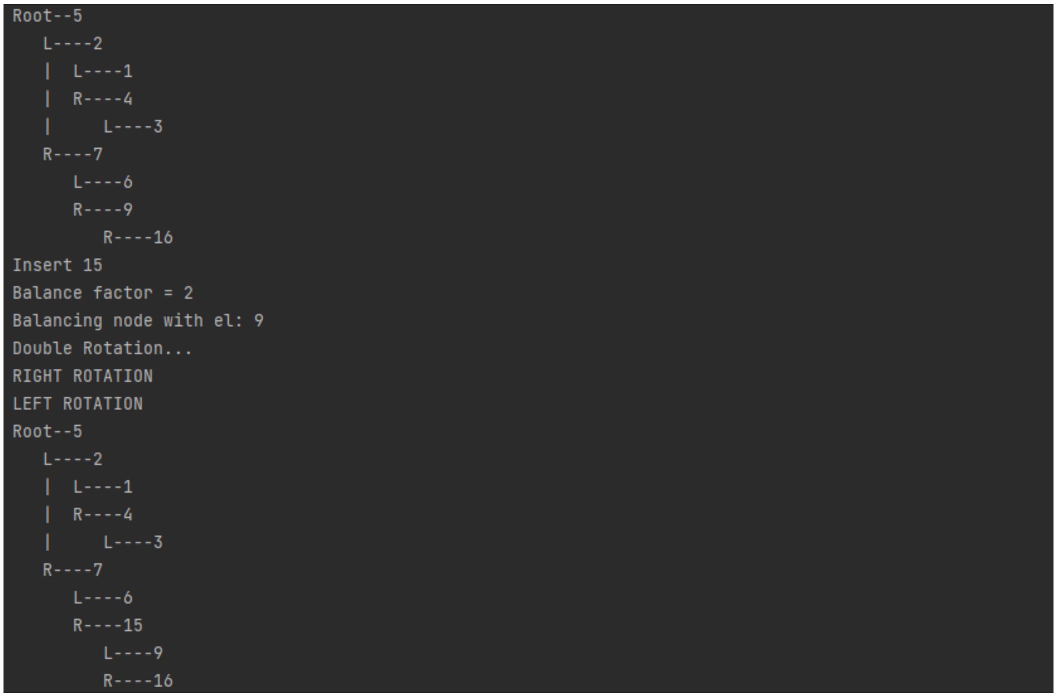


```java
protected void rotateRight() {
    System.out.println("RIGHT ROTATION");
    // to be completed by students
    // store right node
    BTNode<T> tempNode = root.right;
    // change the right of the root to left
    root.right = root.left;
    // change the left of the root to its left of the left
    root.left = root.left.left;
    root.right.left = root.right.right;
    // return the original right to be the right of the right root
    root.right.right = tempNode;

    // swap between root data and rights data
    T val = (T) root.data;
    root.data = root.right.data;
    root.right.data = val;

    getRightAVL().adjustHeight();
    adjustHeight();
}
```



```java
protected void rotateLeftRight()
{
    System.out.println("Double Rotation...");
    getLeftAVL().rotateLeft();
    getLeftAVL().adjustHeight();
    this.rotateRight();
    this.adjustHeight();
}
```

Output:

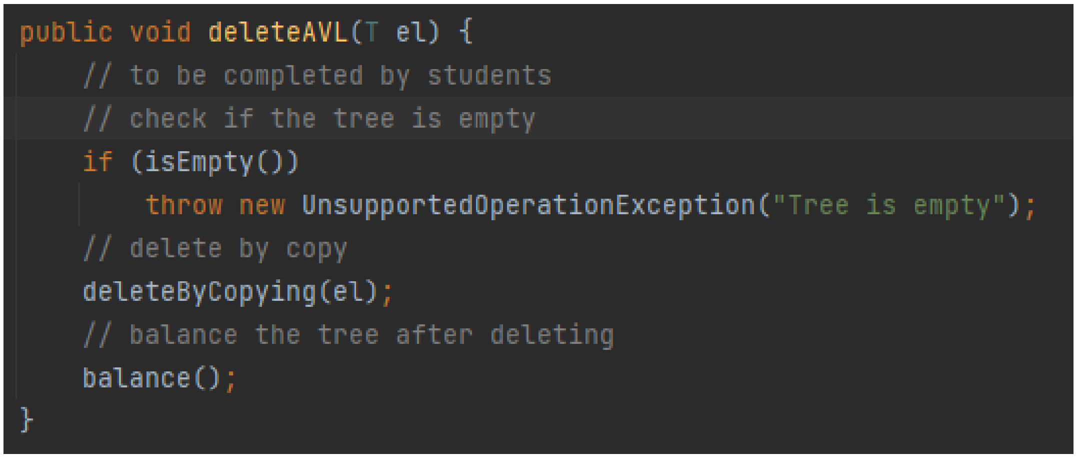


```
Root--5
   L----2
   |  L----1
   |  R----4
   |     L----3
   R----7
      L----6
      R----9
         R----16
Insert 15
Balance factor = 2
Balancing node with el: 9
Double Rotation...
RIGHT ROTATION
LEFT ROTATION
Root--5
   L----2
   |  L----1
   |  R----4
   |     L----3
   R----7
      L----6
      R----15
         L----9
         R----16
```

Task 2: deleteAVL(T el):

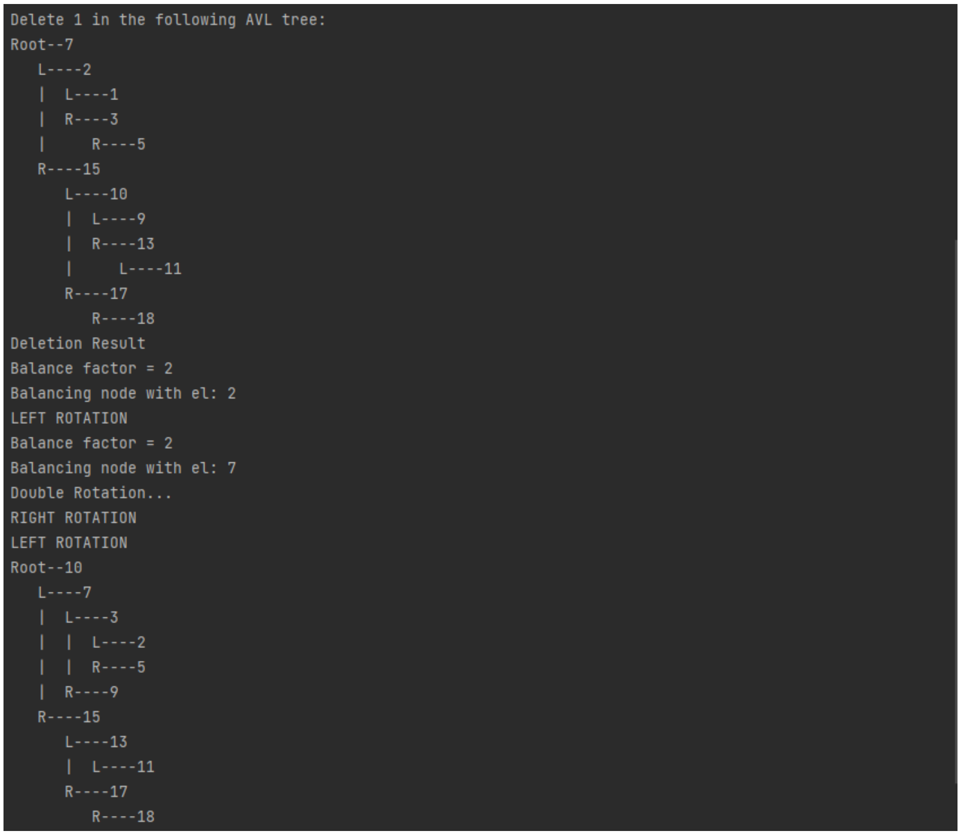
First check if the tree is empty, then delete using deleteByCopying Method from BST, then balance

Code:



```
public void deleteAVL(T el) {
    // to be completed by students
    // check if the tree is empty
    if (isEmpty())
        throw new UnsupportedOperationException("Tree is empty");
    // delete by copy
    deleteByCopying(el);
    // balance the tree after deleting
    balance();
}
```

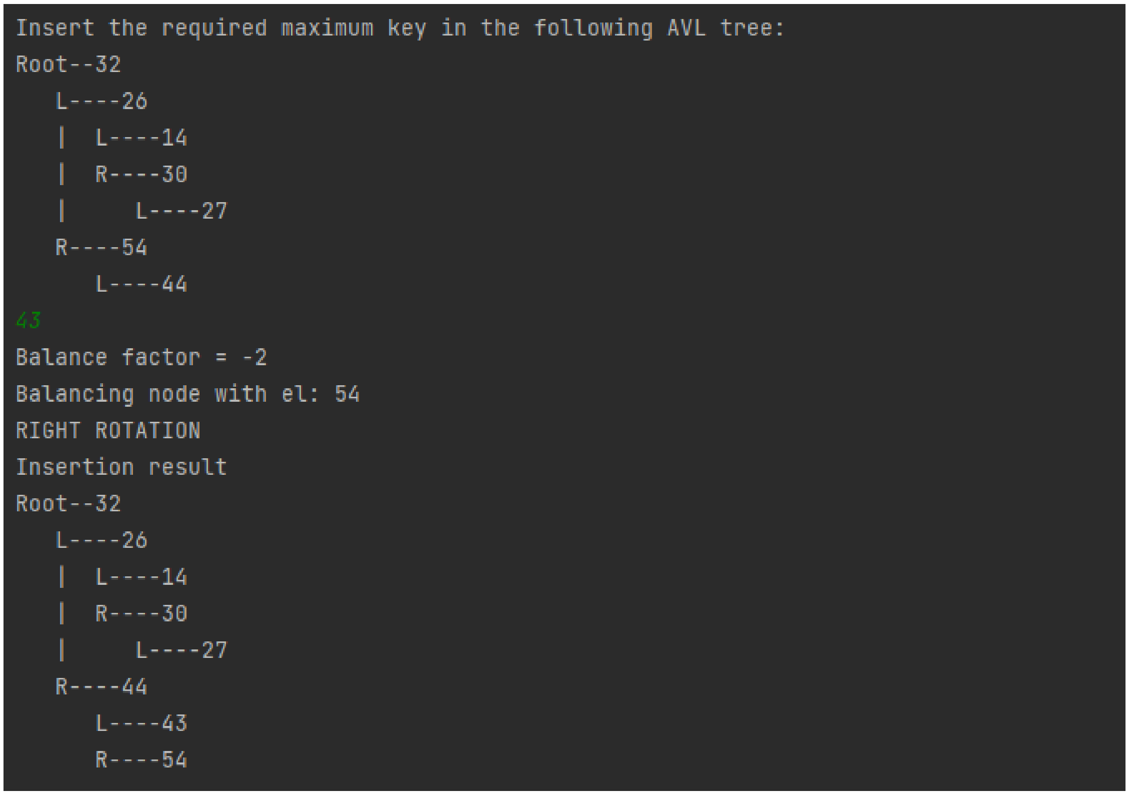
Output:



```
Delete 1 in the following AVL tree:
Root--7
   L----2
   |  L----1
   |  R----3
   |     R----5
   R----15
      L----10
      |  L----9
      |  R----13
      |     L----11
      R----17
         R----18
Deletion Result
Balance factor = 2
Balancing node with el: 2
LEFT ROTATION
Balance factor = 2
Balancing node with el: 7
Double Rotation...
RIGHT ROTATION
LEFT ROTATION
Root--10
   L----7
   |  L----3
   |  |  L----2
   |  |  R----5
   |  R----9
   R----15
      L----13
      |  L----11
      R----17
         R----18
```

Task 3: Find maximum integer key that when inserted in the AVL tree below will cause a single right rotation:

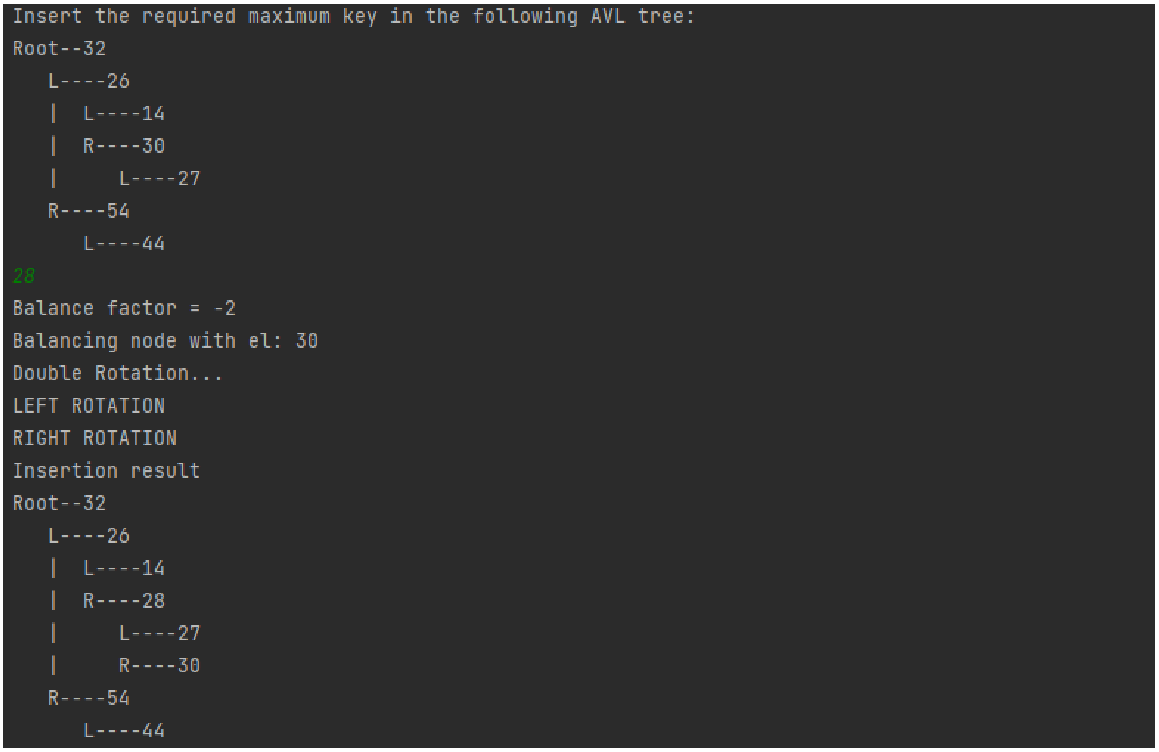
Ans : 43



```
Insert the required maximum key in the following AVL tree:
Root--32
   L----26
   |  L----14
   |  R----30
   |     L----27
   R----54
      L----44
43
Balance factor = -2
Balancing node with el: 54
RIGHT ROTATION
Insertion result
Root--32
   L----26
   |  L----14
   |  R----30
   |     L----27
   R----44
      L----43
      R----54
```

Task 4: Find minimum integer key that when inserted in the AVL tree below will cause a left right rotation:

Ans : 28



```
Insert the required maximum key in the following AVL tree:
Root--32
   L----26
   |  L----14
   |  R----30
   |     L----27
   R----54
      L----44
28
Balance factor = -2
Balancing node with el: 30
Double Rotation...
LEFT ROTATION
RIGHT ROTATION
Insertion result
Root--32
   L----26
   |  L----14
   |  R----28
   |     L----27
   |     R----30
   R----54
      L----44
```