Here some of my e-mails replies:

* Do not make a change on AVLNode class
* count in AVLNode class denotes number of nodes with same key in the tree
* noOfNodes in AVL class denotes number of nodes with unique key in the tree

**Check Test 1 method :**

avl.Insert(10);

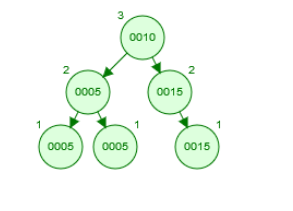
avl.Insert(5);

avl.Insert(15);

avl.Insert(15);

avl.Insert(5);

avl.Insert(5);



This is how the tree looks like after insertions

Apparently, nodes with same keys added to right, if you insert them to left, please change (it can't be that hard you just have to replace “<=” with “<” )

node = avl.Root();

**if** (node.key != 10 || node.count != 1) **return** 0;

node = avl.Root().left;

**if** (node.key != 5 || node.count != 3) **return** 0;

node = avl.Root().right;

**if** (node.key != 15 || node.count != 2) **return** 0;

Root node is 10 and, number of nodes that’s key is 10 is 1.   
Left node of the root 5 and there is 3 node with the key 5.

Right node of the root is 15 and there is 2 node with the key 15.

**Check Test2 method**

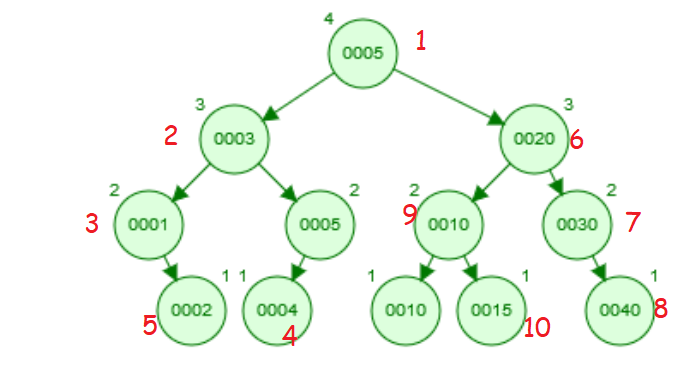
|  |  |
| --- | --- |
| avl.Insert(5);  avl.Insert(5);  avl.Insert(10);  avl.Insert(10);  avl.Insert(1);  avl.Insert(3);  avl.Insert(2);  avl.Insert(4);  avl.Insert(20);  avl.Insert(15);  avl.Insert(30);  avl.Insert(40); | This is how the tree looks like after insertions |

There is two 5 and two 10, thus, number of unique keys 10 (noOfNodes = 10)

I’m updating the test class as “depth != 3” (sorry, my mistake)

// Check the validity of the tree

|  |  |
| --- | --- |
| 1 | node = avl.Root();  **if** (node.key != 5 || node.count != 2) **return** 0; |
| 2 | node = avl.Root().left;  **if** (node.key != 3 || node.count != 1) **return** 0; |
| 3 | node = avl.Root().left.left;  **if** (node.key != 1 || node.count != 1) **return** 0; |
| 4 | node = avl.Root().left.right.left;  **if** (node.key != 4 || node.count != 1) **return** 0; |
| 5 | node = avl.Root().left.left.right;  **if** (node.key != 2 || node.count != 1) **return** 0; |
| 6 | node = avl.Root().right;  **if** (node.key != 20 || node.count != 1) **return** 0; |
| 7 | node = avl.Root().right.right;  **if** (node.key != 30 || node.count != 1) **return** 0; |
| 8 | node = avl.Root().right.right.right;  **if** (node.key != 40 || node.count != 1) **return** 0; |
| 9 | node = avl.Root().right.left;  **if** (node.key != 10 || node.count != 2) **return** 0; |
| 10 | node = avl.Root().right.left.right;  **if** (node.key != 15 || node.count != 1) **return** 0; |



**Checking Test3 method**

|  |  |
| --- | --- |
| avl.Insert(5);  avl.Insert(5);  avl.Insert(10);  avl.Insert(10);  avl.Insert(1);  avl.Insert(3);  avl.Insert(2);  avl.Insert(4);  avl.Insert(20);  avl.Insert(15);  avl.Insert(12);  avl.Insert(11);  avl.Insert(13);  avl.Insert(17);  avl.Insert(16);  avl.Insert(18);  avl.Insert(30); | After insertion |

**if** (avl.Delete(35) == 0) **return** 0;

no “35” in the tree, so Delete returns -1

**if** (avl.Delete(30)!=0) **return** 0;

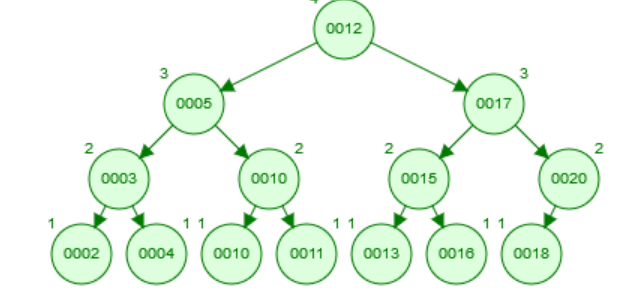
30 is a leaf node and deleting it does not cause an imbalance

|  |  |
| --- | --- |
| node = avl.Find(20);  **if** (node.right!=**null** && node.left.key!=18)  **return** 0; |  |

|  |  |
| --- | --- |
| avl.Delete(1);  node = avl.Find(3);  **if** (node.left.key != 2 && node.right.key != 5)  **return** 0; |  |

|  |  |
| --- | --- |
| avl.Delete(5);  node = avl.Find(3);  **if** (node.left.key != 2 && node.right.key != 4)  **return** 0; |  |

Q : There is two 5, which one should be deleted ?   
A: The one which you found first, in this case, the root will be deleted. It wiil be replaced with largest value if left subtree, by just luck it’s the other 5. If you replace with smallest value of right subtree, please change as such. Here how the tree should like after Delete(5)



|  |  |
| --- | --- |
| avl.Delete(15);  node = avl.Find(13);  **if** (node.right.key != 12 && node.right.key != 17) **return** 0; |  |

**Mistake ! Correct form :**

|  |
| --- |
| **if** (node.right.key != 16) **return** 0; |

|  |  |
| --- | --- |
| avl.Delete(12);  node = avl.Root();  **if** (node.key != 11)  **return** 0;  **if** (avl.NoOfNodes()!=11) **return** 0; |  |

There is 12 node in the tree but two of them are 10, so “NoOfNodes = 11”

**In order to simplification, I’m removing Test 4 and changing grades of others Test methods as 30 points per each (Total 90 and 10 points for report, 100 total )**

* Deadline for the homework is 7th January till midnight, and we have to submit all homework grades to OgrBilSis before 8th January, so there is no time for objection for this homework. So please double check to submit your homework to Mergen, and if you encounter any problem, do not hesitate to contact teaching assistant
* Only zip your **report** and AVL class code (**AVL.java**) in a file which you name as Group\_XX (you group number), I have to evaluate all homeworks in a few hours, thus, please do not make it confusing.
* Your grade will be result after running test class + grade for report (except for the 3 groups who are already submitted, unless they resend )
* Check the rotations from

<https://www.cs.usfca.edu/~galles/visualization/AVLtree.html>

* Term project for these groups are missing in Mergen

|  |  |  |
| --- | --- | --- |
| **Group 9** | İbrahim Kaan | ERDAŞ |
| Enes | GILBAZ |
| Kerem Emre | TUNÇ |
| **Group 13** | Ali Osman | Ünaldı |
| Dilara | Çağlayan |
| Murat Can | Koçyiğit |
| **Group 30** | Furkan | Çakır |
| Doğanay | Tay |
| Helin | Aydemir |
| **Group 31** | Anıl Kaan | Kahveci |
| Baran | Filimci |
| Noor | Mohammed |
| **Group 34** | Hidayet | Yıldırım |
| İrem Canan | Üstününal |
| İbrahim | Kul |
| **Group 36** | Murathan | Kılıç |
| Bora | Algan |
| Zeki Volkan | Ün |

Make sure to you submitted your work, if you had a problem while uploading, please contact teaching assistant   
  
Sincerely