# CS 531: Computer Systems and Fundamentals of Systems Programming
# Instructor: Hal Greenwald
# Homework 4 with Rubric – Due Nov. 10th, 11:59pm

Homework # 4 builds upon Homework # 3.

You will convert the BST implemented in Homework # 3 to an **AVL tree**.

Your program will appear the same as HW#3 to the user *except* that the "Display list" menu option will additionally display the *balance factor* for each node within the tree. The Root of the tree will be indicated as such. Recall from class discussion that the **balance factor** of a node is the absolute difference in height between its left and right child nodes (i.e. subtrees).

**Program structure and design:**

- displayList(), displayAliasesForLocation(), and saveFile() shall be based on **Inorder Traversal**
- displayList() will display the alias, address, height, depth, balance factor, and parent's alias for each node.
- deleteAddress(), lookUpAddress(), and displayAliasesForLocation() will display an error message if the alias (or location) entered is not listed. Following the error message, the menu will be redisplayed.
- A separate UDF will be defined for each menu option.
- No duplicate aliases or address are allowed. If attempted, display an appropriate error message followed by the menu.
- For this exercise, all aliases will be entered in lower case.

In addition to the UDFs contained within HW#3, you should also include the following UDFs (you may design your own function prototypes).

- rebalance(Node n)
- rotateLeft(Node n)
- rotateRight(Node n)
- rotateLeftThenRight(Node n)
- rotateRightThenLeft(Node n)

**Rubric (10 points)**

- Is the source code well documented and formatted using clearly readable indentation and white space (while viewed within **vi**)? **1 point**
- Is the AVL Tree and associated recursion properly implemented? **2 points**
- displayList() must display the correct alias, address, height, depth, balance factor, and parent's alias for each node in order to receive credit. **2 points**

- Does each menu option map to its own UDF, and is each UDF properly implemented? **5 points**
- 1 global variable may be used. Each additional global variable **will cost 1 point**
- **Note:** Your program must compile using gcc/unix in order to receive credit.