

## Quiz – 5

Time allowed: 10 min

### Balanced BSTs, AVL Trees

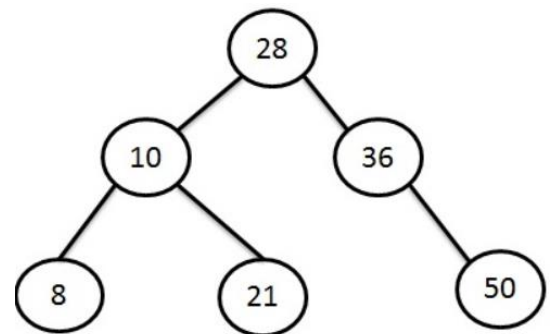
Name: \_\_\_\_\_

ID: \_\_\_\_\_

**Refer to the given AVL tree for questions 2-3.**

**Q1)** What is the worst-case time complexity of deletion in an AVL tree? [1.5 pts]

**Q2)** State one element that, if inserted into the given tree, will require a left rotation. [1.5 pts]



**Q3)** Construct the final updated AVL tree after performing the following operations, sequentially: [3 pts]

`insert(9), delete(10)`

**Q4)** Complete conditions 1 and 2 in the given code to update an AVL tree after deleting a node. Assume you have a helper function `getBalance(Node* node)` that returns the balance factor by subtracting the height of the right subtree from the height of the left subtree. [4 pts]

```
Node* rebalance(Node* root) {  
    if ( root == nullptr)  
        return root;  
    updateHeight(root);  
    int bal = getBalance(Node* root);  
    // Condition 1  
    if  
  
        return leftRotate(root);  
    // Condition 2  
    if  
  
        return leftRightRotate(root);  
}
```