CS 202

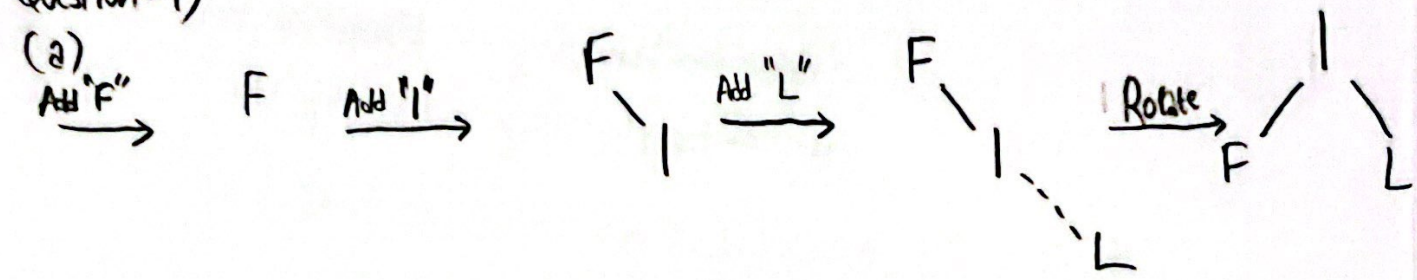
Homework 3

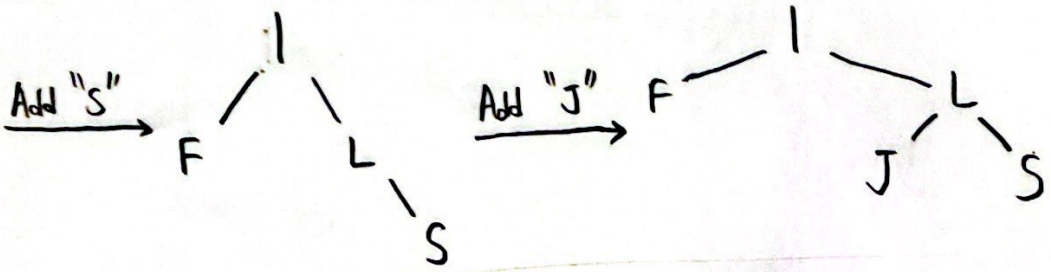
Mehmet Feyyaz Küçük
22003550

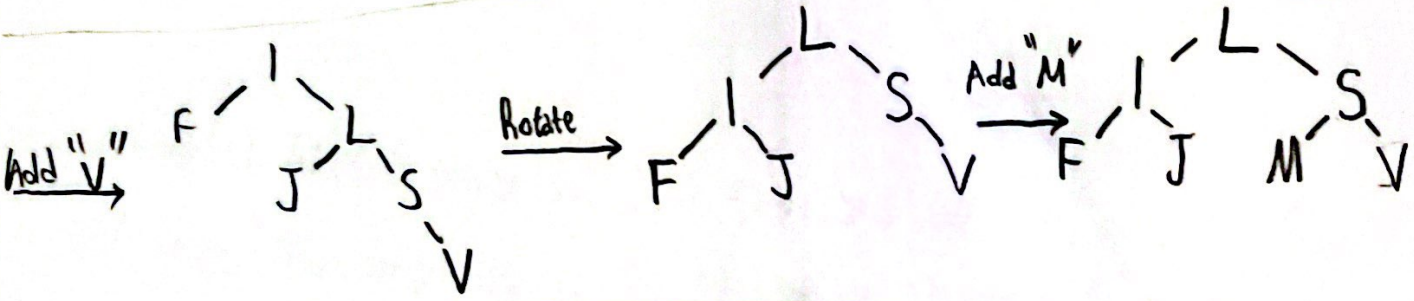
19.4.2022

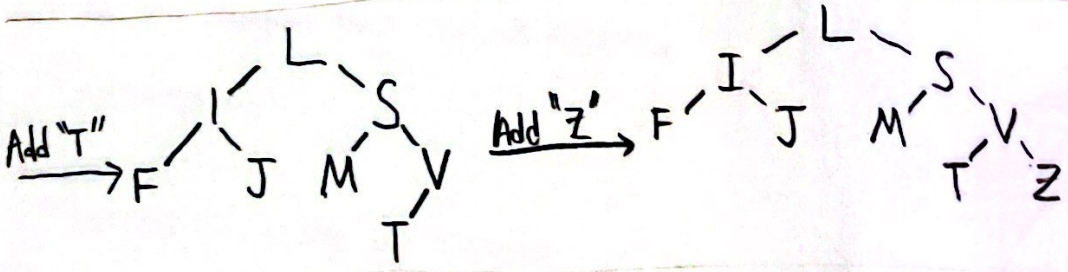
Question-1)

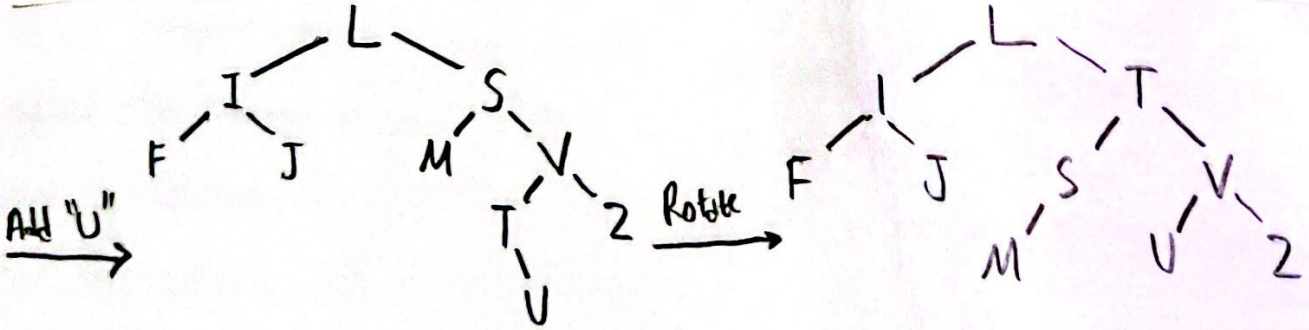
(a)

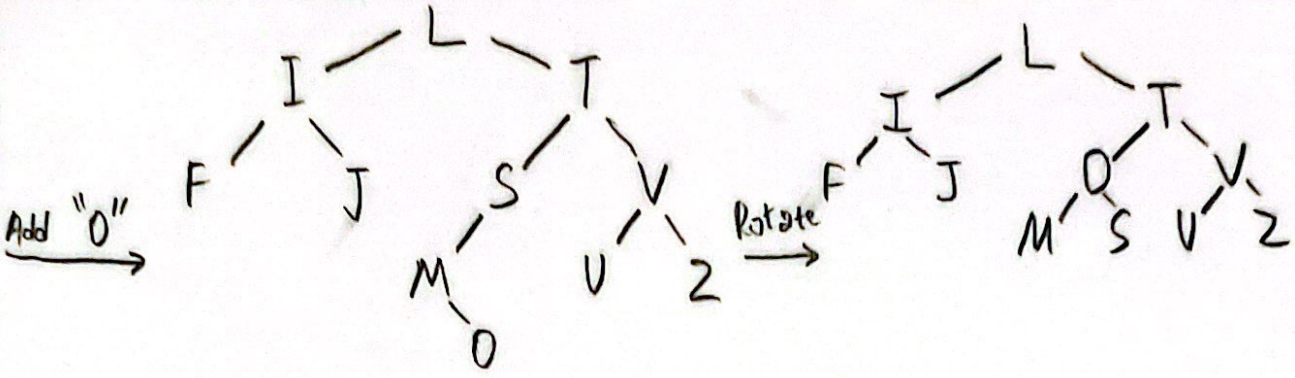












(b)

```
double computeMedian(Node* root) {
    int[] roots;
    while (root -> left -> size != root -> right -> size) {
        // add root to roots
        remove(root);
    }
    // add the balanced root to roots.
    double median = (median of roots);
    // Median of the AVL tree is equal to the median
    // of the roots array.

    return median;
}
```

Removing from AVL tree works in O(logn). Finding the median of the array is O(1) time. Hence, the overall complexity of computeMedian function is O(logn). If the size of left and right subtrees are equal, root is the median because median is the (n/2+1)th element and root has n/2 elements in the right and left subtree. If not remove root until the left and right subtrees have the same size.

(c)

```
bool checkAVL(Node* root) {
    if (root == NULL) {
        return true;
    }

    int leftHeight = getHeight(root->left);
    int rightHeight = getHeight(root->right);

    if ( abs(leftHeight - rightHeight) <= 1 && checkAVL(root->left)
        && checkAVL(root -> right) {
        return true;
    }

    return false;
}
```

getHeight() function works in O(n) time and we calculate height for all nodes. The complexity is $O(n^2)$. We get left and right height of each node and compare their absolute difference.

Question-3)

Instead of adding one everytime, we can multiply the number of computers by 2 everytime starting from 1. This will make it much faster, but we will exceed the minimum amount, which can create a problem. So, when the average waiting time is achieved, we can now start decrementing until the given average waiting time gets close to what we are calculating.