Data Structures and Algorithms

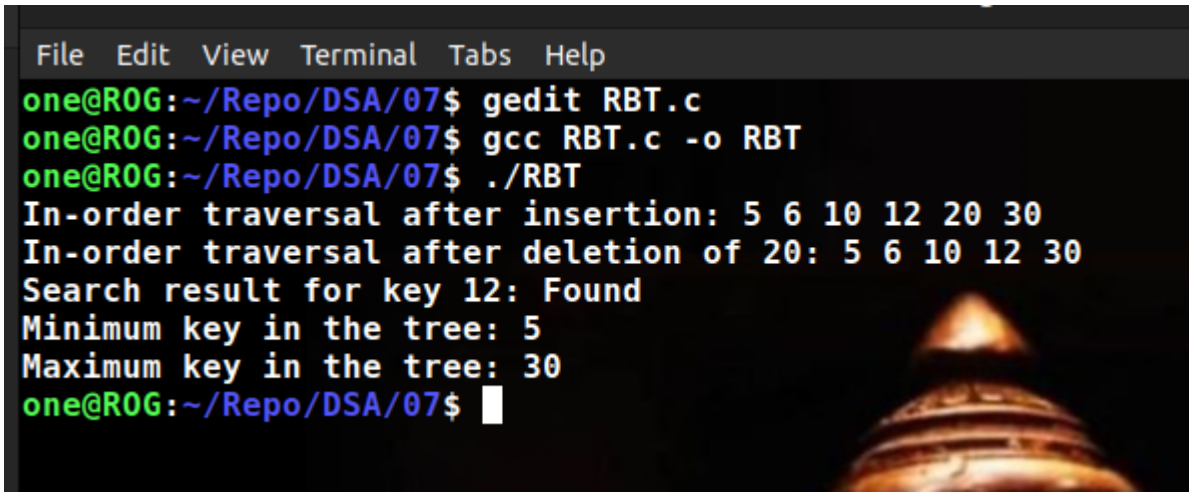
Assignment - VII

Insertion and Validation:
Deletion:
Search and Minimum/Maximum:
In-order Traversal:

INPUT:

```c
int main() {
    struct Node *root = NULL;

    // Insertion and Validation
    printf("Insertion and Validation:\n");
    int keys[] = {7, 3, 18, 10, 22, 8, 11, 26, 2, 6, 13};
    int numKeys = sizeof(keys) / sizeof(keys[0]);

    for (int i = 0; i < numKeys; ++i) {
        insert(&root, keys[i]);
        printf("Inserted %d: ", keys[i]);
        inOrderTraversal(root);
        printf("\n");
    }

    // Deletion
    printf("\nDeletion:\n");
    int deleteKeys[] = {3, 11, 18, 10, 22, 7};
    int numDeleteKeys = sizeof(deleteKeys) / sizeof(deleteKeys[0]);

    for (int i = 0; i < numDeleteKeys; ++i) {
        struct Node *nodeToDelete = search(root, deleteKeys[i]);
        if (nodeToDelete != NULL) {
            printf("Deleting %d: ", deleteKeys[i]);
            deleteNode(&root, nodeToDelete);
            inOrderTraversal(root);
            printf("\n");
        } else {
            printf("%d not found in the tree. Skipping deletion.\n", deleteKeys[i]);
        }
    }

    // Search, Minimum, and Maximum
    printf("\nSearch, Minimum, and Maximum:\n");
    int searchKeys[] = {8, 13, 2, 18, 7};
    int numSearchKeys = sizeof(searchKeys) / sizeof(searchKeys[0]);

    for (int i = 0; i < numSearchKeys; ++i) {
        struct Node *result = search(root, searchKeys[i]);
        if (result != NULL) {
            printf("Key %d found in the tree.\n", searchKeys[i]);
        } else {
            printf("Key %d not found in the tree.\n", searchKeys[i]);
        }
    }

    printf("Minimum key: %d\n", minimumKey(root)->key);
    printf("Maximum key: %d\n", maximumKey(root)->key);

    // In-order Traversal
    printf("\nIn-order Traversal:\n");
    inOrderTraversal(root);
    printf("\n");
```

OUTPUT:


File Edit View Terminal Tabs Help
one@ROG:~/Repo/DSA/07$ gedit RBT.c
one@ROG:~/Repo/DSA/07$ gcc RBT.c -o RBT
one@ROG:~/Repo/DSA/07$ ./RBT
In-order traversal after insertion: 5 6 10 12 20 30
In-order traversal after deletion of 20: 5 6 10 12 30
Search result for key 12: Found
Minimum key in the tree: 5
Maximum key in the tree: 30
one@ROG:~/Repo/DSA/07$