CSE 301 – Algorithm Analysis

Name : Özgür CANSIZ
Student ID : 20190808035

In this assignment, I employed the Java programming language to showcase my solution.

The Implementation:

```
public class FindMajority {

    // Helper method to check if x is the majority element in A
    private static boolean isMajority(int[] A, int x) {
        int count = 0;
        for (int num : A) {
            if (num == x) {
                count++;
            }
        }
        return count > A.length / 2;
    }

    // Recursive method to find the majority element in the array
    private static int findMajorityElement(int[] A, int left, int right) {
        if (left == right) {
            return A[left];
        }

        int mid = (left + right) / 2;

        int leftMajority = findMajorityElement(A, left, mid);
        int rightMajority = findMajorityElement(A, mid + 1, right);

        // Check if leftMajority or rightMajority is the majority element
        if (isMajority(A, leftMajority)) {
```

```
            return leftMajority;
        } else if (isMajority(A, rightMajority)) {
            return rightMajority;
        }

        return -1; // No majority element found
    }

    // Method to initiate the finding of the majority element
    public static int majorityElement(int[] nums) {

        return findMajorityElement(nums, 0, nums.length - 1);

    }

    // Example usage
    public static void main(String[] args) {
        int[] arr = {3, 3, 4, 2, 4, 4, 2, 4, 4}; // Example array
        int majority = majorityElement(arr);
        System.out.println("Majority element is: " + majority);
    }
}
```

Space Compexity:

The space complexity of this algorithm is O(log n) due to the recursive calls made in the findMajorityElement() method, where the recursive stack occupies memory proportional to the logarithm of the input size n.

Time Complexity:

The time complexity of this algorithm is O(n log n), where n is the size of the input array. This is because the algorithm divides the array into halves recursively (log n), and at each level, it checks the majority elements of the left and right halves (n).