Asymptotic Analysis Assignment Report

Question 1

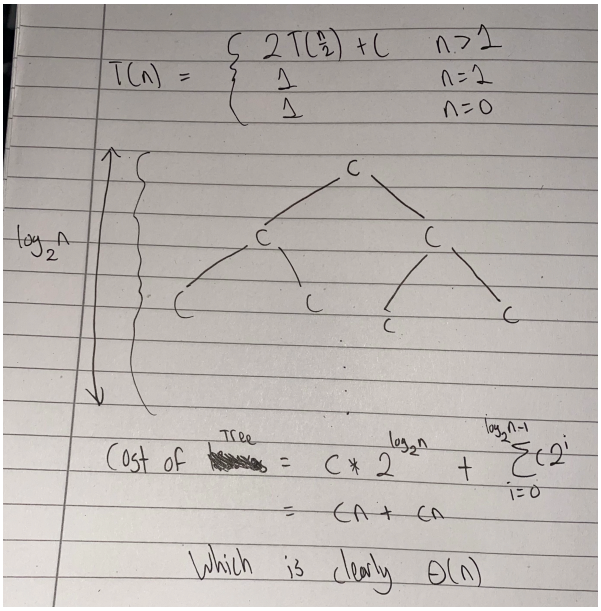
a) Program Code

```
//Question 1

public static int powerIterative(int a,int n){
    int ans = 1;  1
    for(int i=1;i<=n;i++){  n+1
        ans*=a;  n
    }
    return ans;  1
}

public static int powerRecursive(int a,int n){
    if (n==0) return 1;
    if (n==1) return a;
    if(n%2==0)return powerRecursive(a, n: n/2) * powerRecursive(a, n: n/2);
    else return powerRecursive(a,(n/2) ) * powerRecursive(a, n: (n/2)+1);
}
```

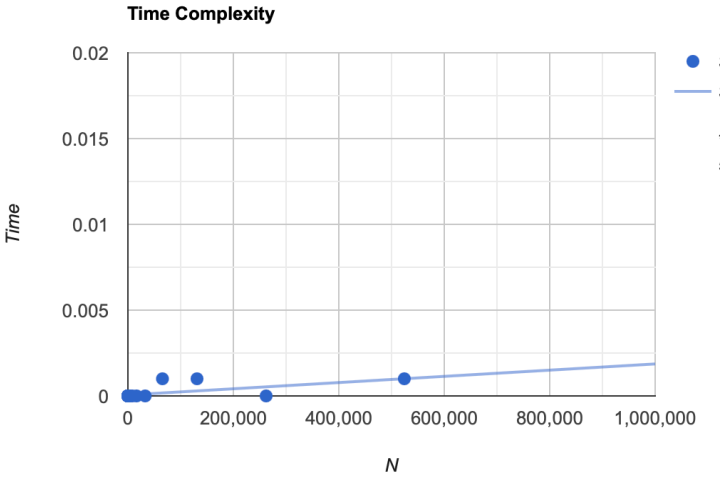
b) Determining Asymptotic Time Complexity

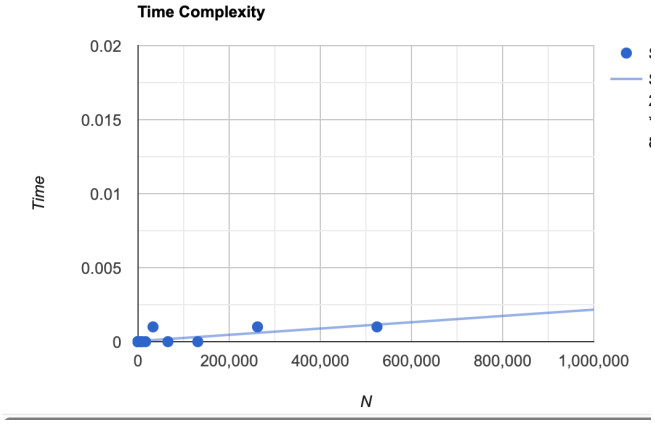
For the iterative method we can clearly see the number of steps taken from the diagram, the number of total steps will be 2n+3 which is θ(n).

For the divide-and-conquer algorithm, solving the recurrence relation:



c)





d) Clearly the experimental results confirm the theoretical analysis results in 1.(b).

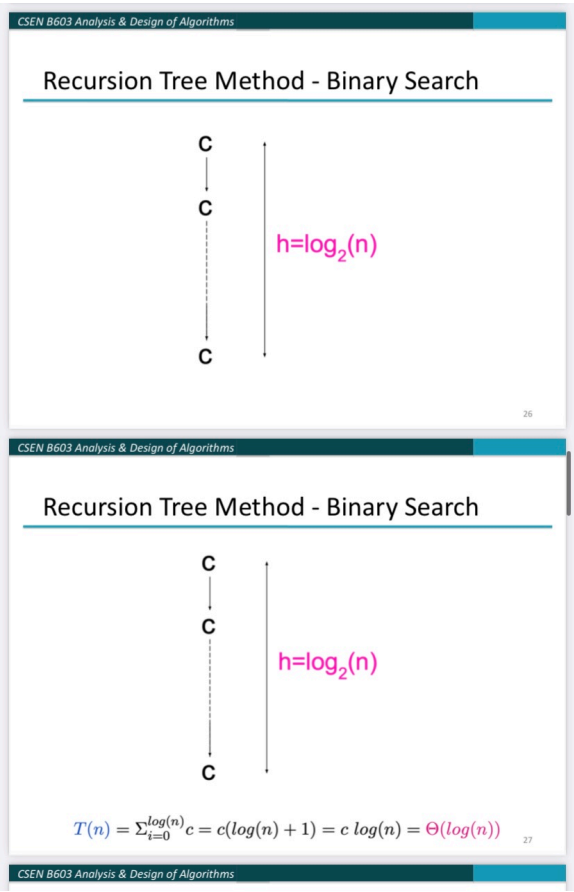
Question 2

a) Program Code

```
public static List<int[]> findPairsWithSum(int[] set, int target) {
    List<int[]> result = new ArrayList<>();


    mergeSort(set, left: 0, right: set.length - 1);

    for (int i = 0; i < set.length; i++) {
        int complement = target - set[i];
        if (binarySearch(set, complement, left: i + 1, right: set.length - 1))
            result.add(new int[]{set[i], complement});
        }
    }

    return result;
}
```

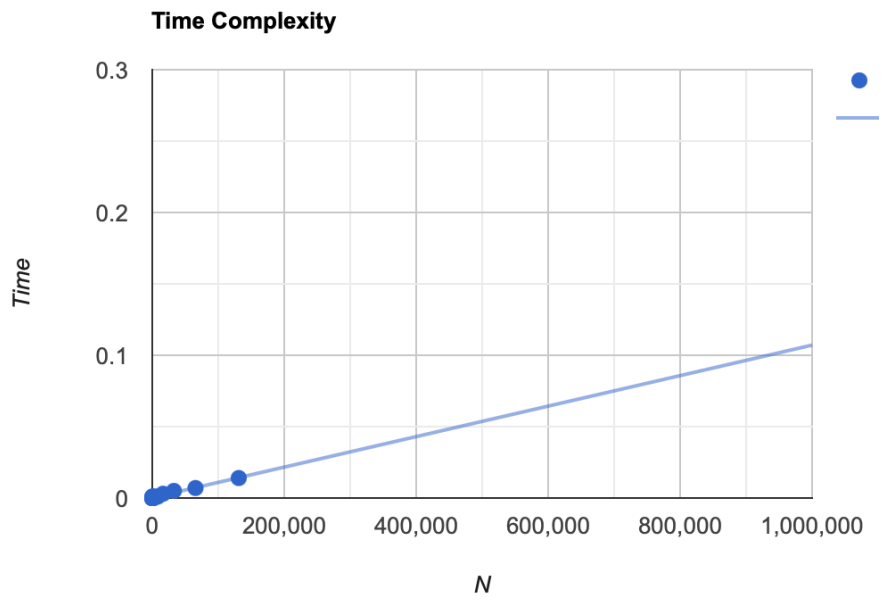
(Check repo for rest of code)

b) The recurrence relation is essentially that of binary search:



The final time complexity will actually be (n * log(n) + log(n)) since Merge Sort is done once and binary search is done n times. Overall θ(nlog(n)).

c)



D) And again the experimental results confirm the theoretical analysis results in 2.(b).