**Lab-4: 19CSE212 Data Structure and Algorithms**

**Implementation of Recursive Algorithms**

1. **Tower of Hanoi (TOH) Problem :**

**Input: n disks stacked over a tower in ascending order of their diameters. (Source Tower)**

**Output: The same arrangement is to be completely shifted to another tower. (Destination Tower)**

**Objective: Minimize the number of moves while shifting the disks.**

**Constraints:**

* **The disks must be placed in the destination tower in the same order as they are stacked in the source tower.**
* **In one move, only 1 disk can be shifted.**
* **At any time, a larger disk can not be placed over a smaller one.**

**Pseudocode:**

**TOH(int n, int L, int M, int R)**

**if n != 0**

**TOH(n-1, L, R, M)**

**Print: Move from L to R**

**TOH(n-1, M, L, R)**

**main()**

**{**

**TOH(5, 1, 2, 3)**

**Assignments**

1. **Analyse the recursive TOH algorithm to compute its time complexity. Hint: T(n) = T(n-1) + T(n-1) +1, (use back-substitution method)**
2. **Implement the recursive algorithm to print the moves for n =5**
3. **Binary Search :**

**pre-condition: take a sorted list of elements as input.**

**Pseudocode:**

**BSearch(A, Low, High, search.key)**

**{**

**if Low < High**

**Mid =**

**if search.key = = Mid**

**Print: value found at Mid**

**return Mid**

**else if search.key > Mid**

**Bsearch(A, Mid+1, High)**

**else Bsearch (A, Low, Mid)**

**}**

**Assignments**

1. **Analyse the recursive binary search algorithm to compute its best-case and worst-case time complexity.**
2. **Implement the recursive algorithm to search an element in a given sorted list of numbers.**
3. **Design and implement an iterative algorithm for the binary search and analyse its time complexity.**
4. **Merge Sort :**

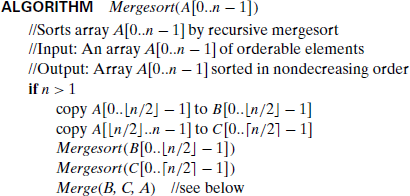
**pre-condition: Use the Divide-Conquer-Combine approach.**

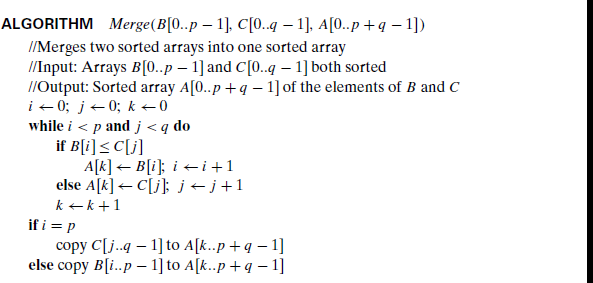
**Divide: Divide the problem into smaller sub-problems.**

**Conquer: Recursively solve the smaller sub-problems.**

**Combine: Combine the results of the smaller sub-problems to compute the result for the larger problem.**

**Pseudocode:**

****

****

**Assignments**

1. **Analyse the recursive merge sort algorithm to compute its time complexity.**
2. **Implement the recursive algorithm to sort a given list of numbers in ascending order.**
3. **Make necessary changes in the existing code to sort a given list of numbers in descending order.**