Insertion Sort Algorithm

Problem Definition:

Implement Insertion Sort algorithm and determine the time required to sort the elements.

Repeat the experiment for different values of N, the number of elements in the list to be sorted and plot a graph of the time taken versus N.

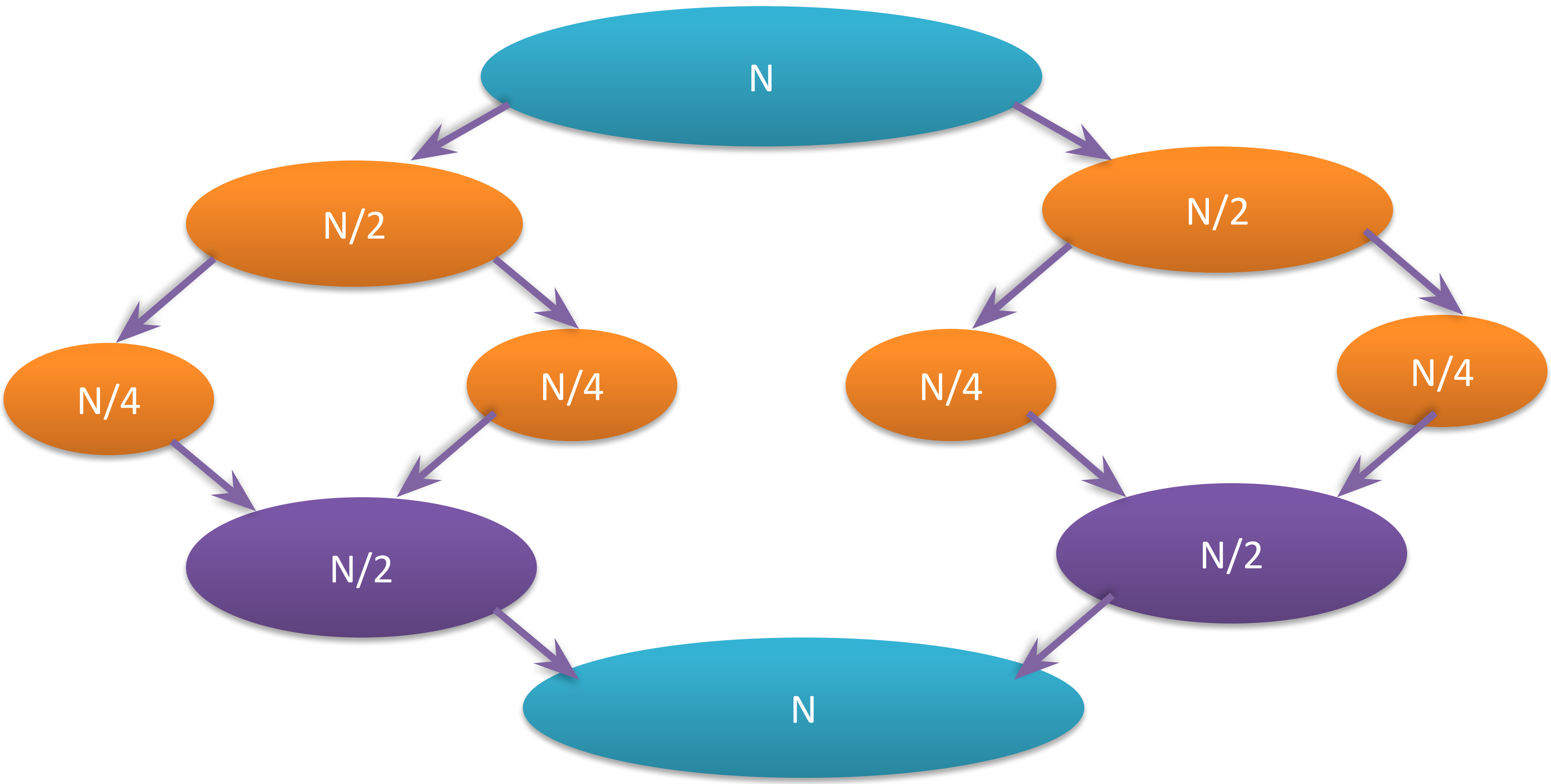
Objectives of the Experiment:

1. To introduce the Decrease & Conquer strategy

2. Present the working of Insertion Sort

3. Analyze the Algorithm & Estimate computing time

Concept : Divide and Conquer - Problem Solving Design strategy



Theoretical Background of the Experiment

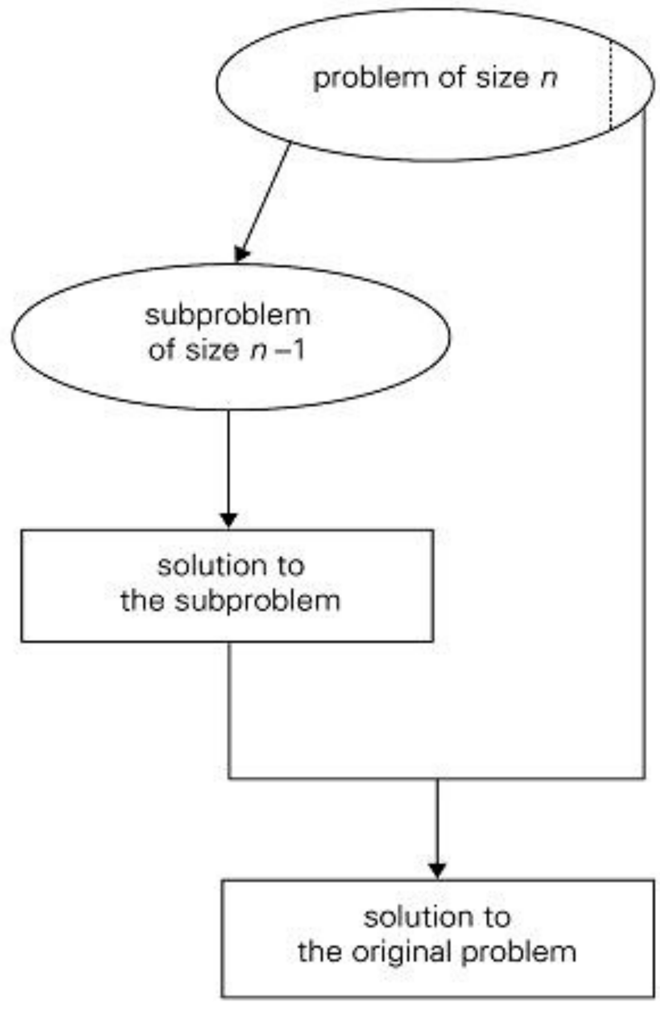
Design strategy - Decrease and Conquer

Decrease and Conquer algorithm make the problem smaller by reducing problem at each step.

Basic idea:

- Reduce problem instance to smaller instance
- Solve smaller instance
- The solution to this smaller instance, would form the basis to larger instances of the same problem.

Theoretical Background of the Experiment

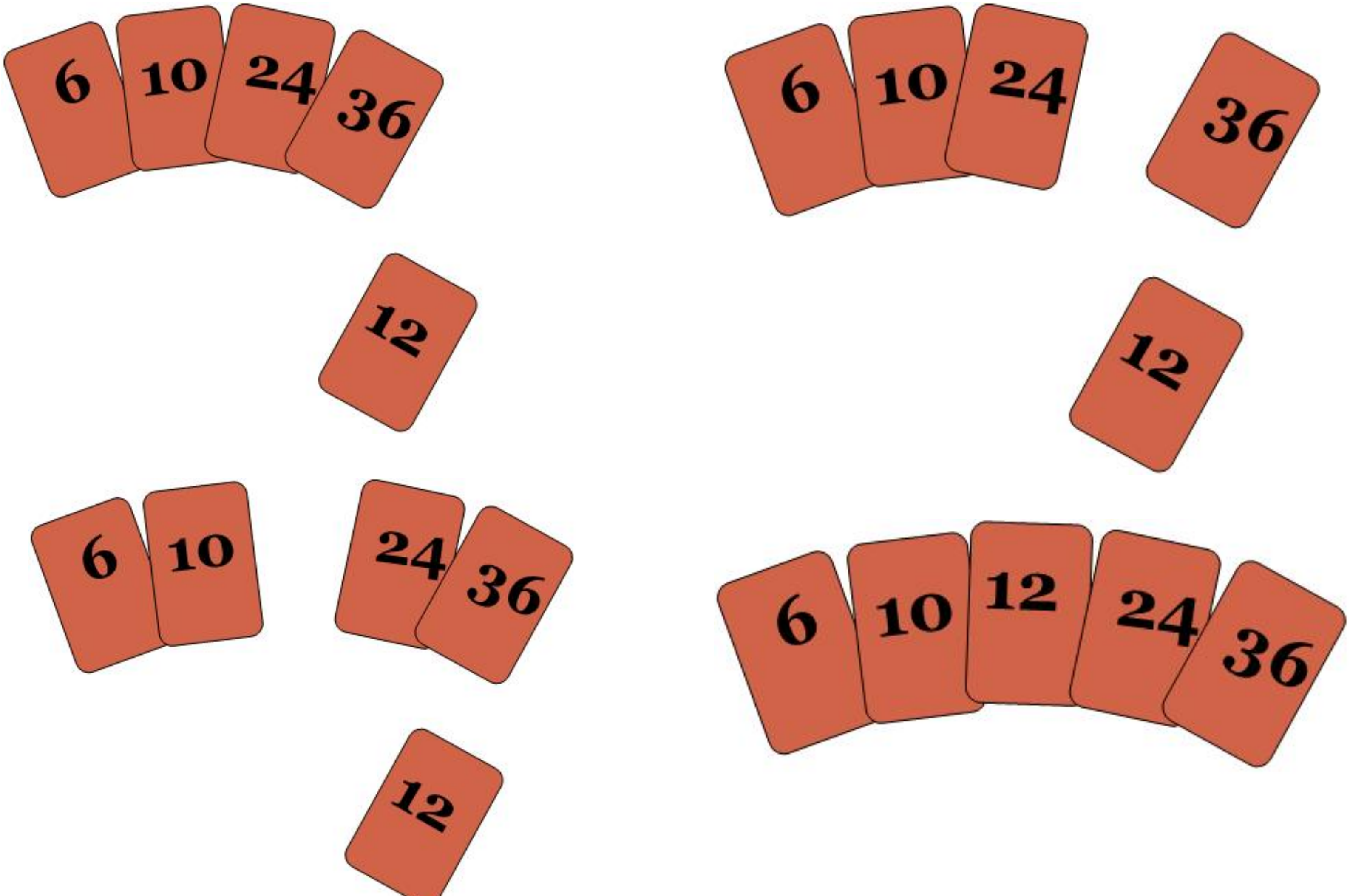


Insertion Sort - Working

- Real life example:

An example of an insertion sort occurs in everyday life while playing cards.

- Start with an empty left hand and cards face down on the table.
- Then remove one card at a time from the table and Insert it into the correct position in the left hand.
- To find a correct position for a card, we compare it with each of the cards already in the hand from right to left .
- This process is repeated until all the cards are in the correct sequence.



Insertion Sort works in the same way.

Example of Insertion Sort

Let us take one example and see the Operations of Insertion Sort on the given **Array**.

A= { 8 , 2, 4, 9, 3, 6}

Example of Insertion Sort

8 2 4 9 3 6 (n-1)

The sorted array is built one element at a time.

Example of Insertion Sort

8 2 4 9 3 6

Example of Insertion Sort

8	2	4	9	3	6	
2	8	4	9	3	6	(n-2)

Example of Insertion Sort

8 2 4 9 3 6

2 8 4 9 3 6

Example of insertion sort

8	2	4	9	3	6	
2	8	4	9	3	6	
2	4	8	9	3	6	(n-3)

Example of Insertion Sort

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

Example of Insertion Sort

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6 (n-4)

Example of Insertion Sort

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6

Example of Insertion Sort

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6
2	3	4	8	9	6

Example of Insertion Sort

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6

2 3 4 8 9 6 (n-5)

Example of Insertion Sort

8	2	4	9	3	6	
2	8	4	9	3	6	
2	4	8	9	3	6	
2	4	8	9	3	6	
2	3	4	8	9	6	
2	3	4	6	8	9	*done*

Insertion-Sort (A , n)

Input : a sequence of *n* numbers stored in array, A.

Output : an ordered sequence of *n* numbers.

```
INSERTION-SORT (A, n)    ▻ A[0 . . n-1]
    for  i ← 1 to n  // process array elements one @ a time
            key ← A[i]  // key – element to be inserted in its correct position
                j ← i – 1
            while    j >= 0 and A[j] > key
                          A[j+1] ← A[j]
                  j ← j – 1
              end while
             A[j+1] = key // place the key in its correct position
    end for
```

// identifies the correct position of the “key” in each pass

Initial Call: Insertion-Sort(*A*, n-1)

Sample Input / Output or Test Cases

Sample 1 : n = 9

4, 2, 7, 1, 9, 0, 3, 8, 11

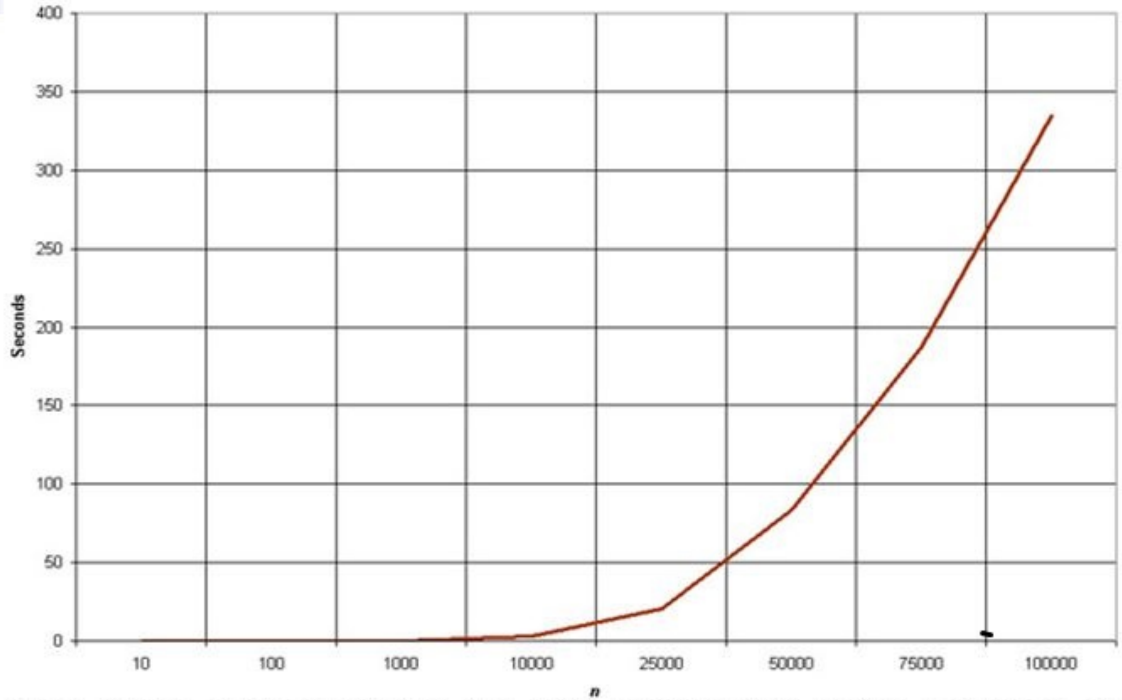
Sample 2 : n = 10

9, 8, 7, 6, 5, 4, 3, 2, 1, 0

Generate the list using Random Function

Time Complexity : $T(n) = O(n^2)$

Empirical Analysis of Insertion Sort



The graph demonstrates the n^2 complexity of the insertion sort.

Learning Outcome of the Experiment and Conclusion

At the end of the session, students should be able to :

1. Explain the working of Decrease and Conquer Strategy
2. Demonstrate the working of Insertion Sort algorithm on a given set of size n
3. Write the program in Java to implement Insertion Sort Algorithm and estimate the computing time using appropriate time functions.
4. Plot a graph of Computing time V/s Size of the input and draw conclusions