Analysis & design of Algorithms

ASSIGNMENT-2

Q.1 Problem Statement:

Implement a **Hybrid Quick Sort algorithm** with the following conditions:

1. Use **Quick Sort** normally.
2. If the sub-array size becomes ≤ 10, switch to **Insertion Sort**.
3. Use **Median-of-Three method** for pivot selection.
4. Count:
    - Number of comparisons
    - Number of swaps
5. Test the program for:
    - Sorted array
    - Reverse sorted array
    - Random array
6. Compare execution time with normal Quick Sort.

Expected Output:

- Sorted array
- Total comparisons
- Total swaps
- Execution time

Q.2 Write a program to:

1. Implement both:
    - Standard Matrix Multiplication
    - Strassen's Matrix Multiplication
2. Accept matrix size **n × n**, where n is a power of 2.
3. Compare:
    - Number of scalar multiplications
    - Execution time
4. Show experimentally that Strassen performs fewer multiplications than standard method.
5. Plot performance for n = 2, 4, 8, 16.

Q.3 **Problem Statement:**

Design a program to solve the following variation:

1. You are given:
    - Profit array
    - Weight array
    - Capacity W
2. You must:

- First apply **Fractional Knapsack**
- Then verify whether solution remains optimal if items are restricted to 0/1.

3. Show clearly where Greedy fails.
4. Compare result with Dynamic Programming solution.

Q.4 **Problem Statement:**

Implement **Kruskal's Algorithm** with the following condition:

1. Some edges are marked as **mandatory**.
2. The MST must include all mandatory edges.
3. If mandatory edges form a cycle, print:
   `"No feasible MST exists"`
4. Print:
   - Total cost
   - Edges in MST
   - Time complexity analysis

Q.5 Write a program to:

- Find minimum and maximum element of an array using Divide & Conquer.
- Count number of comparisons.
- Compare with simple linear approach.
