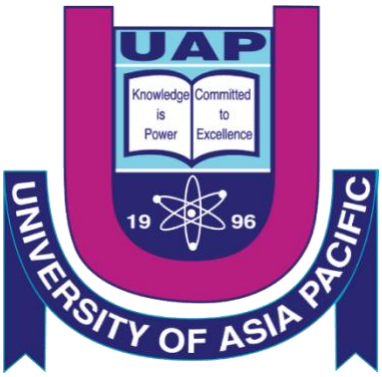

University Of Asia Pacific

Department of CSE

Course Code : **CSE 208**

Course Title : **Data Structures and Algorithms II Lab**

No Of Assignment : **05**

Assignment Name : **Discuss the basic differences between the recursive and dynamic programming approach.**

Date of Submission : **14.11.2024**

Submitted By:

Name : **Nusrat Ahmmed Ekra**

Student Id : 22201251

Section : E2

Semester : 2nd Year 2nd Semester

Submitted To:

Suri Dipannita Sayeed

Lecturer,

CSE,UAP

Problem:02:

Discuss the basic differences between the recursive approach and dynamic programming approach.

Solution:

The differences between the recursive approach and the dynamic programming (DP) approach to solving the 0/1 Knapsack problem:

Feature	Recursive Approach	Dynamic Programming (DP) Approach
Approach to Problem Solving	Solves by breaking the problem into subproblems, explores all possible subsets of items by including or excluding each item.	Solves by breaking the problem into subproblems and storing results in a table (array) to avoid redundant calculations, using a bottom-up approach.
Time Complexity	Exponential $O(2^n)$, inefficient due to redundant calculations.	O(n* W) where n is the number of items and W is the knapsack capacity, much more efficient.
Space Complexity	O(n) due to function call stack in recursion.	O(n* W) for the 2D table, but can be optimized to O(W).
Redundancy in Calculations	High redundancy, recalculates the same subproblems multiple times.	Low redundancy, stores results of subproblems and avoids recalculation.
Readability and Simplicity	Simple, intuitive, and easy to understand for small inputs.	More complex, involves managing a table and iterating over it, harder to grasp initially.
Optimality and Practical Use	Not suitable for large inputs due to exponential time complexity, works for small-scale problems.	Optimal and efficient, suitable for real-world problems, especially with larger datasets.
Examples of Use	Useful for small instances, educational purposes, or algorithm demonstrations.	Practical for large datasets, real-world applications where efficiency is crucial.