Assignment No.04

Title of the Assignment:
Write a program to solve a 0-1 knapsack problem using dynamic programming of branch & bound strategy.

Objective of the Assignment:
students should be able to understand & solve 0-1 knapsack problem using dynamic programming.

Prerequisite :

1. Basic of Python or Java Programming.
2. Concept of Dynamic Programming.
3. 0-1 knapsack problem.

Contents for Theory:

1. Greedy Method
2. 0/1 knapsack Problem.
3. Example solved using 0/1 knapsack problem.

Theory :

What is Dynamic Programming ?

- Dynamic Programming is also used in optimization problems. Like divide-and-conquer method, Dynamic Programming solves problem by combining the solutions of subproblem.

- Two main properties of a problem suggest that the given problem can be solved using dynamic Programming. These properties are overlapping sub-problems & optimal substructure.

- For example, Binary search does not have overlapping sub-problem. Whereas recursive program of Fibonacci num have many overlapping sub-problem.

Steps of Dynamic Programming Approach :

- Characterize the structure of an optimal solution.

- Recursively define the value of an optimal solution.

- Compute the value of an optimal solution, typically a bottom-up fashion.

- Construct an optimal solution from the computed inform

Applications of Dynamic Programming Approach :

- Matrix chain Multiplication

- Longest common subsequence

- Travelling Saleman Problem

```
# code
# A Dynamic Programming b
# Program for 0-1 Knapsack
# Returns the maximum val
# be put in a knapsack of ca
def knapSack(W, wt, val, n)
dp = [0 for i in range(W+1)
for i in range(1, n+1):
# Making the dp array
# taking first i elements
for w in range(W, 0, -1):
have data of
# starting from back.so th
# previous computation w
items
if wt[i-1] <= w:
# finding the maximum v
dp[w] = max(dp[w], dp[
return dp[W]
# returning the maximur
# Driver code
val = [60, 100, 120]
wt = [10, 20, 30]
W = 50
n = len(val)
print(knapSack(W, wt,
Output
220
```

Knapsack Problem :

You are given the following -

- A knapsack with limited weight capacity.
- Few items each having some weight and value

Knapsack Problem Variants :

Knapsack problem has the following two variants :

1. Fractional knapsack Problem
2. 0/1 knapsack Problem.

0/1 knapsack Problem Using Greedy Method .

Consider ,

- knapsack weight capacity = w
- Number of items each having some weight & value = n

Step - 01 :

- Fill all the boxes of 0^{th} row & 0^{th} column with zeroes as shown -

	0	1	2	3	....	W
0	0	0	0	0	....	0
1	0					
2	0					
	....					
n	0					

T - Table

Slep-02:

Use the following formula -

$T(i,j) = \max\{T(i-1, j), value_i + T(i-1, j-weight)\}$

Step-03:

- To identify the items that must be put into the knaps to obtain that maximum profit.

- Consider the last column of the table.

- start scanning the entries from bottom to top.

- After all the entries are scanned, the marked labels represent the items that must be put into the knapsack.

Time Complexity:

- Each entry of the table requires constant time $O(1)$ for computation.

- Overall $O(nw)$ time is taken to solve 0/1 knapsack probl using dynamic programming

Conclusion:

In this way, we have explored concept of 0/1 knaps using dynamic approach.