Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

Artificial Intelligence and Data Science Department.

AOA / Even Sem 2021-22 / Experiment 6.

YASH SARANG.
47 / D6AD.

EXPERIMENT - 6.

Aim: Write a program for the 0/1 Knapsack problem using the Dynamic Programming approach**.**

Theory:

0/1 Knapsack Problem:

Given weights and values of n items, put these items in a knapsack of capacity W to get the maximum total value in the knapsack.
In other words, given two integer arrays
val[0..n-1] and wt[0..n-1]
which represent values and weights associated with n items respectively. Also given an integer W which represents knapsack capacity, find out the maximum value subset of val[] such that sum of the weights of this subset is smaller than or equal to W. You cannot break an item, either pick the complete item or don't pick it (0-1 property).

Complexity Analysis:

Time Complexity:

Normal 1/0 Knapsack: O(2^n)
DPA 1/0 Knapsack: O(n*W),
where ‘N’ is the number of weight elements and ‘W’ is capacity.
As for every weight element we traverse through all weight capacities 1<=w<=W.

Auxiliary Space: O(n*W).
The use of a 2-D array of size ‘n*W’.

Scope for Improvement: We used the same approach as the Normal but with optimized space complexity.

CODE:

Code is in the 0-1-KnapsackDPA.c file attached along with this doc.

INPUT:

```
int val[] = {60, 100, 120};
int wt[] = {10, 20, 30};
int W = 50;
```

OUTPUT:

```
Maximum amount of value we can store : 220
```

CONCLUSION:

In Dynamic Programming(DP) problems, re-computation of the same subproblems can be avoided by constructing a temporary array K[][] in a bottom-up manner and the implementation of the same has helped us in the DPA approach in the 0/1 Knapsack Problem.

By performing this experiment, I can conclude that the time complexity of the Normal approach to the 0/1 Knapsack Problem is exponentially worse than the DPA approach (comparing 2^n to n*W) and it increases the efficiency by a very big margin.
The DPA helps us to avoid certain patterns altogether from the results of the earlier tests, this property makes the algorithm to be favorable in real-time applications.
