**Design and Analysis of Algorithm**

**Experiment No. : 9**

**Write a program to implement a 0/1 knapsack problem using dynamic approach.**

Experiment No. 9

1. **Aim:** Write a program to implement a 0/1 knapsack problem using dynamic approach.
2. **Algorithm**

**0/1 Knapsack Problem Using Dynamic Programming algorithm-**

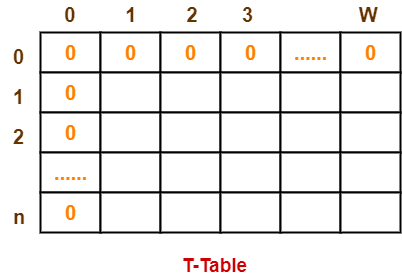
Consider-

* Knapsack weight capacity = w
* Number of items each having some weight and value = n

0/1 knapsack problem is solved using dynamic programming in the following steps-

**Step-01:**

* Draw a table say ‘T’ with (n+1) number of rows and (w+1) number of columns.
* Fill all the boxes of 0th row and 0th column with zeroes as shown-



**Step-02:**

Start filling the table row wise top to bottom from left to right.

Use the following formula-

**T (i , j) = max { T ( i-1 , j ) , valuei + T( i-1 , j – weighti) }**

Here, T(i , j) = maximum value of the selected items if we can take items 1 to i and have weight restrictions of j.

* This step leads to completely filling the table.
* Then, value of the last box represents the maximum possible value that can be put into the knapsack.

**Step-03:**

To identify the items that must be put into the knapsack to obtain that maximum profit,

* Consider the last column of the table.
* Start scanning the entries from bottom to top.
* On encountering an entry whose value is not same as the value stored in the entry immediately above it, mark the row label of that entry.
* After all the entries are scanned, the marked labels represent the items that must be put into the knapsack.
* **Conclusion and Discussion:** Each entry of the table requires constant time θ(1) for its computation. It takes θ(nw) time to fill (n+1)(w+1) table entries. It takes θ(n) time for tracing the solution since tracing process traces the n rows. Thus, overall θ(nw) time is taken to solve 0/1 knapsack problem using dynamic programming.