Knapsack using Dynamic programming

Knapsack problem

Given:

- n items
- Known weight- w1,w2,…wn
- values=- v1,v2,…vn
- Knapsack capacity W

Find:

Most valuable subset of items to fill knapsack

Formula:

V(i,j)

Item i, 1<=i<=n i-1 to n

Weight w1,w2,….wi

Value v1,v2,…vi

Knapsack capacity j 1<=j<=w

Condition

- Subset- ith item doesnot include v[i-1,j]
- Subset- ith item include vi+v[i-1,j-wi]

V[i,j]= max{v[i-1,j], vi+v[i-1,j-wi]}

Formula

$$V[i,j]=\max\{v[I,j-1],v_i+v[i-1,j-w_i]\}$$

	0	J-wi	J	W
0	0	0	0	0
i-1	0	V[i-1,j-wi]	V[i-1,j]	
i	0		V[i,j]	
N	0			Goal

Algorithm

Knapsack(i,j)
Description : implement the knapsack problem
I/P : non negative integer I & j
O/P : optimal feasible subset

```
if v(i,j)<0
   if j<w[i]
      value← knapsack(i-1,j)
       else
       value← knapsack(i-1,j), value(i)+   knapsack(i-1,j-wi)
V(I,j)← value
Return v(I,j)
```

Problem

- Capacity W=5

Items	Weight	Value
1	2	$12
2	1	$10
3	3	$20
4	2	$15

Answer

0	0	0	0	0
0	0	12	12	12
0	10	12	22	22
0	10	12	22	30
0	10	15	25	30