Name: Abhishek

USN NO:2VX23UE001

EXPERIMENT NO: 7a 0/1 Knapsack problem using Dynamic programing

In [3]:
```
weight = [10, 20, 30]
profit = [60, 100, 120]
capacity = 40
n = len(profit)

def DP_KnapSack(capacity, weight, profit, n):
    K = [[0 for x in range(capacity + 1)]  for x in range(n + 1)]

    for i in range(n + 1):
        for w in range(capacity + 1):
            if i == 0 or w == 0:
                K[i][w] = 0
            elif weight [i-1] <= w:
                K[i][w] = max(profit[i-1] + K[i-1][w-weight[i-1]], K[i-1][w])
            else:
                K[i][w] = K[i-1][w]

    return K[n][capacity]

max_profit = DP_KnapSack(capacity, weight, profit, n)

print("maximum profit earned = ", max_profit)
```

```
maximum profit earned =  180
```

In [7]:
```
n = int(input("Enter the number of items : "))

print("Enter the weight and profit for each item : ")
weight = []
profit = []
for i in range(n):
    w = int(input("weight of item {}: ".format(i+1)))
    p = int(input("profit of item {}: ".format(i+1)))
    weight.append(w)
    profit.append(p)

capacity = int(input("Enter the capacity of knapsack : "))

def DP_KnapSack(capacity, weight, profit, n):
    K = [[0 for x in range(capacity + 1)]  for x in range(n + 1)]

    for i in range(n + 1):
        for w in range(capacity + 1):
            if i == 0 or w == 0:
                K[i][w] = 0
```

```
            elif weight [i-1] <= w:
                K[i][w] = max(profit[i-1] + K[i-1][w-weight[i-1]], K[i-1][w])
            else:
                K[i][w] = K[i-1][w]

    return K[n][capacity]

max_profit = DP_KnapSack(capacity, weight, profit, n)

print("maximum profit earned = ", max_profit)
```

```
Enter the weight and profit for each item :
maximum profit earned =  70
```

In [ ]: