```python
def knapsack_dp(W, wt, val, n):
    """A Dynamic Programming based solution for 0-1 Knapsack problem
    Returns the maximum value that can"""
    K = [[0 for x in range(W + 1)] for x in range(n + 1)]

    # Build table K[][] in bottom up manner
    for i in range(n + 1):
        for w in range(W + 1):
            if i == 0 or w == 0:
                K[i][w] = 0
            elif wt[i - 1] <= w:
                K[i][w] = max(val[i - 1] + K[i - 1][w - wt[i - 1]], K[i - 1][w])
            else:
                K[i][w] = K[i - 1][w]
    return K[n][W]


val = [60, 100, 120]
wt = [10, 20, 30]
W = 50
n = len(val)
print("Maximum possible profit =", knapsack_dp(W, wt, val, n))

"""
OUTPUT:

Maximum possible profit = 220
"""

```