```
# A Dynamic Programming based Python
# Program for 0-1 Knapsack problem
# Returns the maximum value that can
# be put in a knapsack of capacity W
def knapSack(W, wt, val, n):
      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]

# Driver program to test above function
val = [60, 100, 120]
wt = [10, 20, 30]
W = 50
n = len(val)
print(knapSack(W, wt, val, n))

# This code is contributed by Bhavya Jain
```