**Assignment No.4:- Program to solve a 0-1 Knapsack problem using dynamic programming or branch and bound strategy**

**Name:-Sameer Narendra Tikare**

**Roll No:-COBB034**

# 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]

def InputList():

    lst = []

    n = int(input("Enter number of elements : "))

    for i in range(0, n):

        ele = int(input())

        lst.append(ele)

    return lst

# Driver code

#val = [60, 100, 120]

val = InputList()

#wt = [10, 20, 30]

wt = InputList()

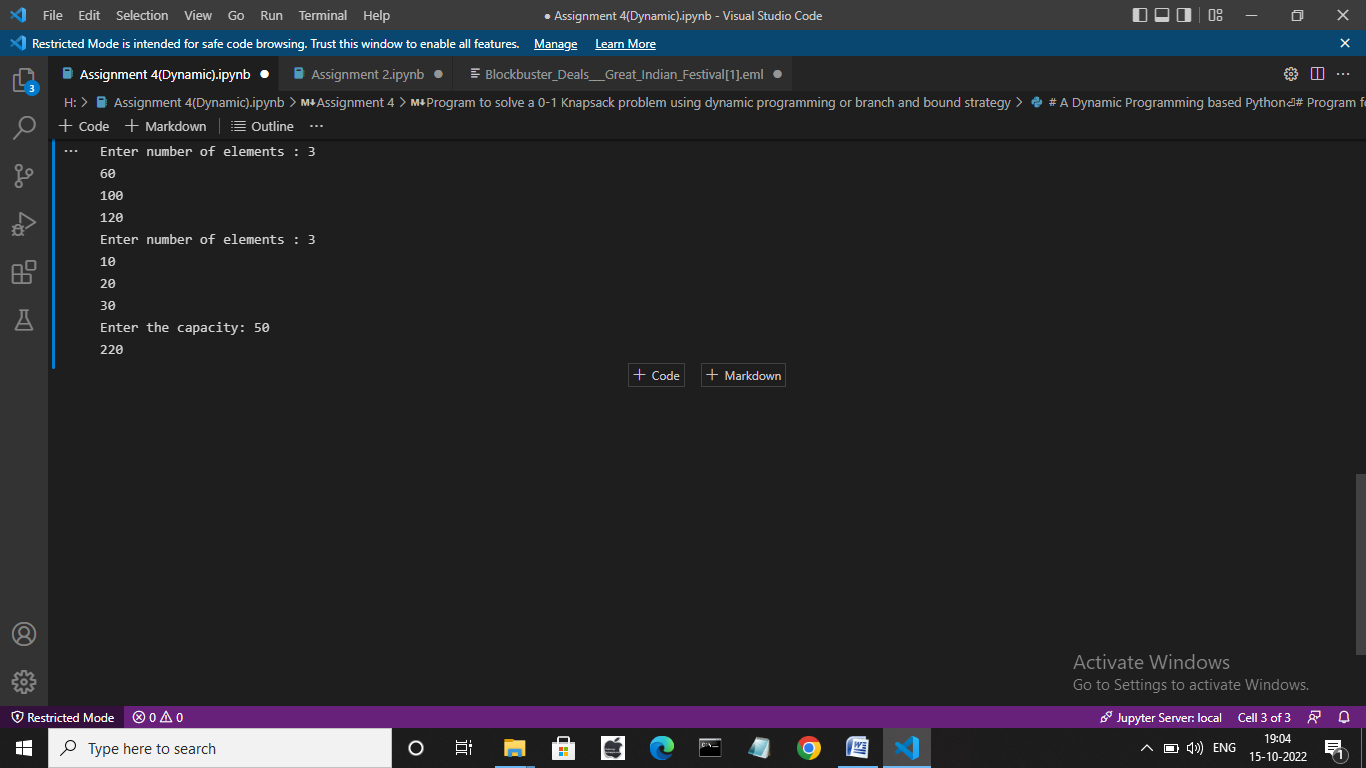
#W = 50

W = int(input("Enter the capacity: "))

n = len(val)

print(knapSack(W, wt, val, n))

**Output**

****