**Name:-Amol Sakharam Gaikwad**

**Roll No:-COBA018**

**CODE**:-

# A naive recursive implementation

# of 0-1 Knapsack Problem

# Returns the maximum value that

# can be put in a knapsack of

# capacity W

def knapSack(W, wt, val, n):

# Base Case

if n == 0 or W == 0:

return 0

# If weight of the nth item is

# more than Knapsack of capacity W,

# then this item cannot be included

# in the optimal solution

if (wt[n-1] > W):

return knapSack(W, wt, val, n-1)

# return the maximum of two cases:

# (1) nth item included

# (2) not included

else:

return max(

val[n-1] + knapSack(

W-wt[n-1], wt, val, n-1),

knapSack(W, wt, val, n-1))

# end of function knapSack

#Driver Code

val = [60, 100, 120]

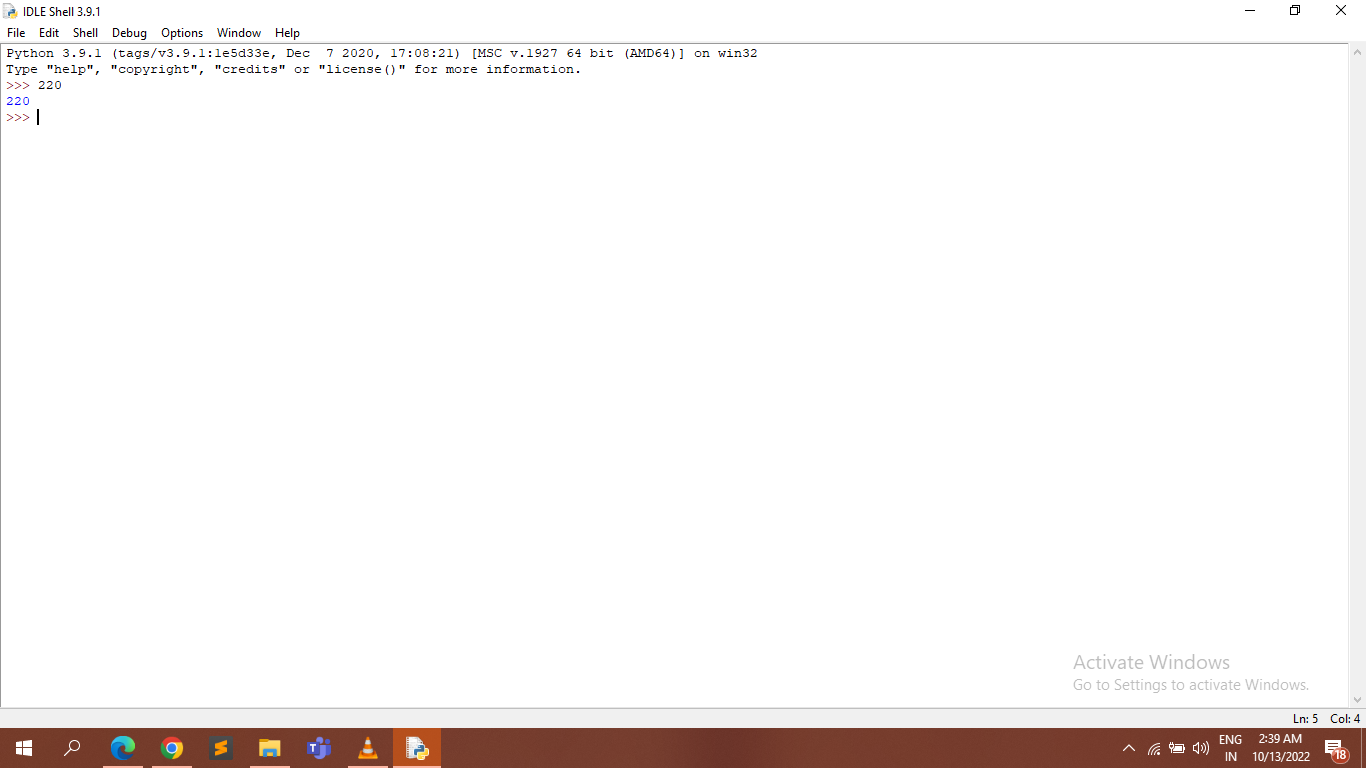
wt = [10, 20, 30]

W = 50

n = len(val)

print knapSack(W, wt, val, n)

**OUTPUT:-**

****