import copy

def sum\_cost(cost):

cost\_sum = copy.deepcopy(cost)

for iobj in range(len(cost) - 2, -1, -1):

cost\_sum[iobj] += cost\_sum[iobj + 1]

return cost\_sum

def complete\_knapsack(cost, value, bagsize):

obj\_num = len(value)

f = [0] \* (bagsize + 1)

cost\_sum = sum\_cost(cost)

for iobj in range(obj\_num):

# ensure the volume of bag can hold the object

bound = max(bagsize-cost\_sum[iobj], cost[iobj])

# jsize changes from bound to bagsize,

# considering each object can be selected more time

for jsize in range(bagsize, bound - 1, -1):

f\_in\_item = f[jsize-cost[iobj]] + value[iobj]

if f\_in\_item > f[jsize]:

f[jsize] = f\_in\_item

return f[1:]

def seperate\_obj(cost, worth, bagsize):

cost\_sep = []

worth\_sep = []

for i in range(len(cost)):

num = 1

cost\_ori = cost[i]

worth\_ori = worth[i]

while cost\_ori \* num < bagsize:

cost\_sep.append(cost\_ori \* num)

worth\_sep.append(worth\_ori \* num)

num \*= 2

return cost\_sep, worth\_sep

bag\_size = 10

cost = [2, 2, 6, 5, 4]

worth = [6, 3, 5, 4, 6]

cost\_sep, worth\_sep = seperate\_obj(cost, worth, bag\_size)

f = complete\_knapsack(cost\_sep, worth\_sep, bag\_size)

print(f)