**Code**

def knapsack\_dynamic\_programming(weights, values, capacity):

num\_items = len(weights)

dp = [[0 for \_ in range(capacity + 1)] for \_ in range(num\_items + 1)]

for i in range(1, num\_items + 1):

for w in range(1, capacity + 1):

if weights[i - 1] <= w:

dp[i][w] = max(values[i - 1] + dp[i - 1][w - weights[i - 1]], dp[i - 1][w])

else:

dp[i][w] = dp[i - 1][w]

# Backtrack to find which items are selected

selected\_items = []

i, w = num\_items, capacity

while i > 0 and w > 0:

if dp[i][w] != dp[i - 1][w]:

selected\_items.append(i - 1)

w -= weights[i - 1]

i -= 1

return dp, dp[num\_items][capacity], selected\_items[::-1]

def main():

num\_items = int(input("Enter the number of items: "))

item\_names = []

weights = []

values = []

for i in range(num\_items):

item\_name = input(f"Enter the name of item {i + 1}: ")

weight = int(input(f"Enter the weight of item {i + 1}: "))

value = int(input(f"Enter the value of item {i + 1}: "))

item\_names.append(item\_name)

weights.append(weight)

values.append(value)

capacity = int(input("Enter the knapsack's capacity: "))

matrix, max\_value, selected\_items = knapsack\_dynamic\_programming(weights, values, capacity)

print("\nKnapsack items selected:")

for index in selected\_items:

print(f"Item name: {item\_names[index]}, Weight: {weights[index]}, Value: {values[index]}")

print("\nMaximum value in the knapsack:", max\_value)

print("\nDynamic Programming Matrix:")

for row in matrix:

print(row)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output**

Enter the number of items: 4

Enter the name of item 1: ItemA

Enter the weight of item 1: 2

Enter the value of item 1: 6

Enter the name of item 2: ItemB

Enter the weight of item 2: 2

Enter the value of item 2: 10

Enter the name of item 3: ItemC

Enter the weight of item 3: 3

Enter the value of item 3: 12

Enter the name of item 4: ItemD

Enter the weight of item 4: 5

Enter the value of item 4: 18

Enter the knapsack's capacity: 5

Knapsack items selected:

Item name: ItemB, Weight: 2, Value: 10

Item name: ItemC, Weight: 3, Value: 12

Maximum value in the knapsack: 22

Dynamic Programming Matrix:

[0, 0, 0, 0, 0, 0]

[0, 0, 6, 6, 6, 6]

[0, 0, 10, 10, 16, 16]

[0, 0, 10, 12, 16, 22]

[0, 0, 10, 12, 16, 22]