def knapsack(weights, values, capacity):

"""

Soluciona el problema de la mochila 0/1 utilizando programación dinámica.

Parameters:

weights (list): lista de pesos de los items

values (list): lista de valores de los items

capacity (int): capacidad máxima de la mochila

Returns:

int: valor máximo que se puede obtener con la capacidad dada

"""

n = len(weights)

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

# Inicializa la matriz dp con ceros

for i in range(n + 1):

for w in range(capacity + 1):

if i == 0 or w == 0:

dp[i][w] = 0

elif weights[i - 1] <= w:

# Si el peso del item es menor o igual que la capacidad,

# se puede elegir entre incluirlo o no

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

else:

# Si el peso del item es mayor que la capacidad,

# no se puede incluir

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

return dp[n][capacity]

# Ejemplo de uso

weights = [2, 3, 5, 7]

values = [10, 20, 30, 40]

capacity = 10

result = knapsack(weights, values, capacity)

print("Valor máximo:", result)

Ejemplos de uso

pesos = [2, 3, 5, 7] valores = [10, 20, 30, 40] capacidad = 10

resultado = mochila(pesos, valores, capacidad) print("Valor máximo:", resultado) explicame como funciona

Explicación del algoritmo de la mochila 0/1

El algoritmo de la mochila 0/1 es un problema clásico de programación dinámica que consiste en encontrar la combinación óptima de elementos que se pueden incluir en una mochila con una capacidad máxima, de manera que se maximice el valor total de los elementos incluidos.

Funcionamiento del algoritmo

La función knapsackrecibe tres parámetros:

weights: una lista de pesos de los artículos

values: una lista de valores de los artículos

capacity: la capacidad máxima de la mochila

La función utiliza una matriz dpde tamaño (n+1) x (capacity+1), donde nes el número de artículos. La matriz se inicializa con ceros.

El algoritmo funciona de la siguiente manera:

Inicialización : se inicializa la matriz dpcon ceros. Esto se hace para evitar errores en la recursividad.

Recursividad : se itera sobre cada elemento y cada capacidad posible. Para cada artículo iy capacidad w, se consideran dos casos:

Si el peso del artículo ies menor o igual que la capacidad w, se puede elegir entre incluirlo o no. En este caso, se calcula el valor máximo que se puede obtener incluyendo el artículo iy el valor máximo que se puede obtener sin incluirlo. Se elige el valor máximo entre estas dos opciones.

Si el peso del artículo ies mayor que la capacidad w, no se puede incluir. En este caso, elija el valor máximo que se puede obtener sin incluir el artículo i.

Retorno : se devuelve el valor máximo que se puede obtener con la capacidad dada, que se encuentra en la celda dp[n][capacity].