109. Knapsack problem

Code:

```
def knapsack(weights, values, capacity):
    n = len(weights)
    dp = [[0] * (capacity + 1) for _ in range(n + 1)]

    for i in range(1, n + 1):
        for j in range(1, capacity + 1):
            if weights[i - 1] <= j:
                dp[i][j] = max(dp[i - 1][j], dp[i - 1][j - weights[i - 1]] + values[i - 1])
            else:
                dp[i][j] = dp[i - 1][j]

    return dp[n][capacity]


weights = [1, 2, 3, 4, 5]
values = [10, 20, 30, 40, 50]
capacity = 7

print("Maximum value:", knapsack(weights, values, capacity))
```

Output:

```
Maximum value: 70
```

Time Complexity:

- T(n)= O(n * capacity)