

Practical- 3 DAA

Name- Pratham Mahabare

Roll No.- 39

Code-

```
def knapsack_01(values, weights, capacity):
    n = len(values)

    # Create a table to store the maximum values for different subproblems
    dp = [[0 for _ in range(capacity + 1)] for _ in range(n + 1)]

    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:
                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 the items selected
    selected_items = []
    i, w = n, 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

    selected_items.reverse()

    return dp[n][capacity], selected_items

# Example usage
if __name__ == "__main__":
    values = [12, 40, 25, 42]
    weights = [3, 4, 5, 7]
    max_capacity = 9

    max_value, selected_items = knapsack_01(values, weights, max_capacity)

    print("Selected Items:")
    for item in selected_items:
        print(f"Item with weight {weights[item]} and value {values[item]}")

    print(f"Maximum value achievable: {max_value}")
```

Output-

```
> ~/Downloads/LP-3/DAA
pratham_mahabare@fedora:~/Downloads/LP-3/DAA$ python3 3_zero_one_knapsack.py
Selected Items:
Item with weight 4 and value 40
Item with weight 5 and value 25
Maximum value achievable: 65
pratham_mahabare@fedora:~/Downloads/LP-3/DAA$
```