

EXPT 7(b): Knapsack problem using Greedy Method

Name: Ganesh K

USN: 2VX23UE015

```
In [5]: 1 weight = [10, 20, 30]
2 profit = [60, 100, 120]
3 capacity = 40
4 n = len(profit)
5
6
7 def greedy_knapsack(capacity, weight, profit, n):
8
9     ratio = [(profit[i] / weight[i], weight[i], profit[i]) for i in range(n)]
10
11     ratio.sort(reverse=True, key=lambda x: x[0])
12
13     total_profit = 0
14     current_weight = 0
15
16     for ratio_value, w, p in ratio:
17         if current_weight + w <= capacity:
18             total_profit = total_profit + p
19             current_weight = current_weight + w
20
21         else:
22             remaining_capacity = capacity - current_weight
23             total_profit = total_profit + ratio_value * remaining_capacity
24             break
25
26     return total_profit
27
28 max_profit = greedy_knapsack(capacity, weight, profit, n)
29
30 print("Maximum Profit Earned (Greedy Approach) = ", max_profit)
```

Maximum Profit Earned (Greedy Approach) = 200.0

```

In [7]: 1 # Taking inputs form user
        2 n = int(input("Enter the number of items : "))
        3
        4 print("Enter the weight and profit for each item:")
        5 weight = []
        6 profit = []
        7 for i in range(n):
        8     w = int(input("Weigth of item {}: ". format(i+1)))
        9     p = int(input("Profit of item {}: ". format(i+1)))
       10     weight.append(w)
       11     profit.append(p)
       12
       13 capacity = int(input("Enter the capacity of the Knapsack: "))
       14
       15 def greedy_knapsack(capacity, weight, profit, n):
       16
       17     ratio = [(profit[i] / weight[i], weight[i], profit[i]) for i in range(n)]
       18
       19     ratio.sort(reverse=True, key=lambda x: x[0])
       20
       21     total_profit = 0
       22     current_weight = 0
       23
       24     for ratio_value, w, p in ratio:
       25         if current_weight + w <= capacity:
       26             total_profit = total_profit + p
       27             current_weight = current_weight + w
       28
       29         else:
       30             remaining_capacity = capacity - current_weight
       31             total_profit = total_profit + ratio_value * remaining_capacity
       32             break
       33
       34     return total_profit
       35
       36 max_profit = greedy_knapsack(capacity, weight, profit, n)
       37
       38 print("Maximum Profit Earned (Greedy Approach) = ", max_profit)

```

```

Enter the number of items : 5
Enter the weight and profit for each item:
Weigth of item 1: 2
Profit of item 1: 20
Weigth of item 2: 3
Profit of item 2: 60
Weigth of item 3: 5

```

Profit of item 3: 50
Weigth of item 4: 8
Profit of item 4: 70
Weigth of item 5: 9
Profit of item 5: 80
Enter the capacity of the Knapsack: 40
Maximum Profit Earned (Greedy Approach) = 280