# CHAITRA SAMANT

# 231070055

# DAA LAB 5A

1. **AIM:**
   Consider a XYZ courier company. They receive different goods to transport to different cities. Company needs to ship the goods based on their life and value. Goods having less shelf life and high cost shall be shipped earlier. Consider list of 100 such items and capacity of transport vehicle is 200 tones. Implement Algorithm for fractional knapsack problem.

2. **PROGRAM:**
   Assumption: Let us take in input as a nested list. The list contains 100 elements, each element have 3 elements of the format: [ Value, Weight, Shelf Life]

   For example consider the dataset given below:

```
dataset = [
    [1390, 16, 5], [-485, 25, 9], [335, 19, 4], [1475, 37, 6], [290, 12, 3],
    [740, 21, 7], [1230, 9, 2], [540, 35, 8], [1550, 10, 10], [870, 29, 3],
    [295, 49, 5], [1165, 7, 6], [390, 18, 9], [785, 27, 7], [975, 45, 4],
    [1455, 8, 8], [605, 6, 2], [695, 54, 6], [615, 28, 9], [730, 4, 5],
    [1350, 11, 7], [570, 16, 8], [1120, 22, 3], [1375, 42, 10], [410, 5, 7],
    [845, 12, 4], [1215, 14, 9], [785, 11, 6], [470, 31, 9], [545, 23, 2],
    [1330, 48, 8], [605, 40, 5], [1485, 21, 7], [910, 15, 4], [765, 9, 10],
    [1345, 27, 3], [425, 46, 8], [825, 8, 6], [1295, 5, 9], [210, 50, 3],
    [1360, 18, 7], [895, 32, 5], [665, 37, 10], [765, 6, 8], [930, 17, 3],
    [1115, 14, 7], [400, 25, 4], [285, 48, 9], [835, 44, 6], [515, 3, 8],
    [1410, 20, 9], [905, 7, 4], [755, 19, 10], [1310, 16, 5], [435, 14, 7],
    [1385, 5, 9], [985, 28, 4], [735, 24, 8], [1495, 35, 3], [765, 42, 9],
    [280, 46, 6], [505, 53, 7], [1190, 55, 10], [895, 3, 6], [685, 39, 8],
    [1025, 13, 7], [780, 56, 9], [765, 16, 5], [1505, 4, 3], [725, 17, 8],
    [1395, 12, 6], [435, 45, 9], [215, 34, 2], [1175, 33, 10], [910, 29, 5],
    [695, 8, 7], [1215, 18, 9], [1325, 11, 6], [535, 52, 4], [1270, 27, 5],
    [755, 41, 8], [1430, 22, 7], [1200, 36, 10], [980, 47, 6], [1540, 34, 3],
    [870, 7, 8], [485, 9, 4], [1255, 43, 9], [735, 32, 6], [585, 13, 10],
    [1010, 50, 7], [1050, 26, 8], [505, 19, 5], [815, 36, 4], [475, 21, 7]
]
```

```
def check(items):
    """
    This function checks if the input given is valid and all conditions are
satisfied

    Arguments:
    items (list): The dataset with all the information

    Returns:
    Errors : (if any) Will display corresponding errors else continue program
execution
    """
    for ind,item in enumerate(items):
        if item[0] < 0:
            print("Price of item can't be negative")
            exit(1)
        if item[1] < 0:
            print("Weight cannot be negative")
            exit(1)
        if item[2] < 0:
            print("Shelf Life cannot be negative")
            exit(1)


def ptow_ratio(items):
    """
    This function calculates price to weight ratio for each element and
appends it to the list of each element and sorts them in order of shelf life
and p/w ratio

    Arguments:
    items (list): Dataset in which we need to implement fractional knapsack

    Returns:
    None
    """
    if not items:
        print("Dataset is Empty")
        exit(1)
    for item in items:
        if item[1] != 0:
            val = round(item[0]/item[1],2)
        else:
            val = 0
        item.append(val)

    items.sort(key=lambda item: (item[2] , -item[-1]))
```

```
def max_benefit(items,weight):
    """
    This function fills the items in the knapsack and calculates maximum
benefit

    Arguments:
    items (list): Sorted Dataset of all items
    weight (int): Maximum weight of the knapsack(truck)

    Returns:
    None
    """
    benefit = 0

    print("Items loaded in the truck")
    for ind,item in enumerate(items):
        if item[1] <= weight:
            print(f"Weight added:{item[1]}/{item[1]}    Shelf
Life:{item[2]}    P/W Ratio:{item[3]} ")
            weight -=item[1]
            benefit += item[-1]*item[1]

        else:
            benefit += weight*item[-1]
            print(f"Weight added:{weight}/{item[1]}    Shelf
Life:{item[2]}    P/W Ratio:{item[3]} ")
            weight=0

        if weight == 0:
            break
    benefit = round(benefit,2)
    print(f"Total Benefit: {benefit}")


#Driver Code
capacity = 200
check(dataset)
ptow_ratio(dataset)
max_benefit(dataset,capacity)
```

## 3. TESTCASES

- **POSITIVE**
  1. Valid TC1

```
Items loaded in the truck
Weight added:5/5    Shelf Life:1    P/W Ratio:189.2
Weight added:9/9    Shelf Life:1    P/W Ratio:95.0
Weight added:8/8    Shelf Life:1    P/W Ratio:84.5
Weight added:22/22    Shelf Life:1    P/W Ratio:39.91
Weight added:25/25    Shelf Life:1    P/W Ratio:37.56
Weight added:27/27    Shelf Life:1    P/W Ratio:29.48
Weight added:20/20    Shelf Life:1    P/W Ratio:19.0
Weight added:13/13    Shelf Life:1    P/W Ratio:15.92
Weight added:47/47    Shelf Life:1    P/W Ratio:13.89
Weight added:24/47    Shelf Life:1    P/W Ratio:6.32
Total Benefit: 6481.45
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

  2. Valid TC2

```
Items loaded in the truck
Weight added:5/5    Shelf Life:1    P/W Ratio:156.0
Weight added:6/6    Shelf Life:1    P/W Ratio:133.33
Weight added:8/8    Shelf Life:2    P/W Ratio:123.75
Weight added:24/24    Shelf Life:2    P/W Ratio:42.71
Weight added:30/30    Shelf Life:2    P/W Ratio:42.17
Weight added:25/25    Shelf Life:2    P/W Ratio:36.8
Weight added:35/35    Shelf Life:2    P/W Ratio:22.43
Weight added:40/40    Shelf Life:2    P/W Ratio:18.62
Weight added:27/55    Shelf Life:2    P/W Ratio:5.64
Total Benefit: 7462.25
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

3. Valid TC3

```
PS C:\Users\Chaitra\OneDrive\Desktop\Programs> python -u "c:\Users\Chaitra\O
py"
Items loaded in the truck
Weight added:30/30    Shelf Life:1    P/W Ratio:32.5
Weight added:41/41    Shelf Life:1    P/W Ratio:17.07
Weight added:5/5    Shelf Life:2    P/W Ratio:164.0
Weight added:15/15    Shelf Life:2    P/W Ratio:52.0
Weight added:19/19    Shelf Life:2    P/W Ratio:43.42
Weight added:36/36    Shelf Life:2    P/W Ratio:38.61
Weight added:42/42    Shelf Life:2    P/W Ratio:13.21
Weight added:12/35    Shelf Life:2    P/W Ratio:5.71
Total Benefit: 6113.15
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

4. Valid TC4

```
PS C:\Users\Chaitra\OneDrive\Desktop\Programs> python -u "c:\Users\Chaitra\O
py"
Items loaded in the truck
Weight added:32/32    Shelf Life:1    P/W Ratio:32.19
Weight added:45/45    Shelf Life:1    P/W Ratio:15.33
Weight added:6/6    Shelf Life:2    P/W Ratio:141.67
Weight added:18/18    Shelf Life:2    P/W Ratio:68.89
Weight added:17/17    Shelf Life:2    P/W Ratio:46.47
Weight added:34/34    Shelf Life:2    P/W Ratio:42.79
Weight added:20/20    Shelf Life:2    P/W Ratio:42.0
Weight added:28/43    Shelf Life:2    P/W Ratio:12.56
Total Benefit: 7246.5
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

5. Valid TC5

```
Items loaded in the truck
Weight added:9/9    Shelf Life:2    P/W Ratio:136.67
Weight added:6/6    Shelf Life:2    P/W Ratio:100.83
Weight added:23/23    Shelf Life:2    P/W Ratio:23.7
Weight added:34/34    Shelf Life:2    P/W Ratio:6.32
Weight added:4/4    Shelf Life:3    P/W Ratio:376.25
Weight added:17/17    Shelf Life:3    P/W Ratio:54.71
Weight added:22/22    Shelf Life:3    P/W Ratio:50.91
Weight added:27/27    Shelf Life:3    P/W Ratio:49.81
Weight added:34/34    Shelf Life:3    P/W Ratio:45.29
Weight added:24/35    Shelf Life:3    P/W Ratio:42.71
Total Benefit: 10059.85
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

- **NEGATIVE**

6. Dataset absent

```
2   dataset = []

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Chaitra\OneDrive\Desktop\Programs> python -u "c:\Users\Chaitra
py"
Dataset is Empty
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

7. Price of an item is negative

```
1   dataset = [
2       [1390, 16, 5], [-485, 25, 9], [335, 19, 4], [1475, 37, 6], [2

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Chaitra\OneDrive\Desktop\Programs> python -u "c:\Users\Chaitra\
py"
Price of item can't be negative
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

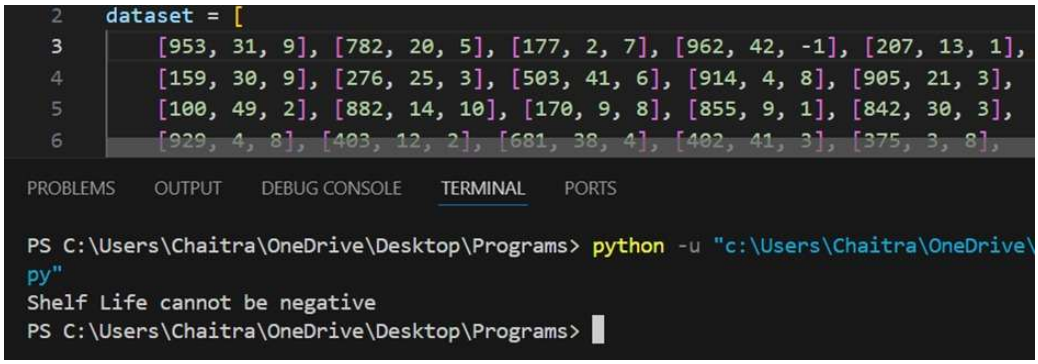
8. Weight of an item is negative

```
2   dataset = [
3       [953, 31, 9], [782, 20, 5], [177, -2, 7], [962, 42, 10], [207, 13, 1],
4       [159, 30, 9], [276, 25, 3], [503, 41, 6], [914, 4, 8], [905, 21, 3],
5       [100, 49, 2], [882, 14, 10], [170, 9, 8], [855, 9, 1], [842, 30, 3],
6       [929, 4, 8], [403, 12, 2], [681, 38, 4], [402, 41, 3], [375, 3, 8],

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Chaitra\OneDrive\Desktop\Programs> python -u "c:\Users\Chaitra\OneDrive\D
py"
Weight cannot be negative
PS C:\Users\Chaitra\OneDrive\Desktop\Programs>
```

9. Shelf Life of an item is negative




**CONCLUSION**

In conclusion, we successfully implemented the fractional knapsack algorithm for the XYZ courier company, optimizing their shipping process by prioritizing goods based on shelf life and value. By ensuring that high-value, short-lifespan goods are shipped first, the company can maximize profits and minimize losses due to expiration. This approach enhances operational efficiency and profits.