

```
Print("ROHAN WAYAL")
```

Class Parcel:

```
Def __init__(self, profit, weight):  
    Self.profit = profit  
    Self.weight = weight  
    Self.ratio = profit / weight if weight != 0 else 0
```

```
Def __repr__(self):  
    Return f"(Profit: {self.profit}, Weight: {self.weight}, Ratio: {self.ratio:.2f})"
```

```
Def fractional_knapsack(capacity, parcels):  
    # Sort parcels by profit/weight ratio in descending order  
    Parcels.sort(key=lambda x: x.ratio, reverse=True)
```

```
Total_profit = 0.0
```

```
Capacity_left = capacity
```

```
Parcels_taken = []
```

```
Print(f"\nTruck capacity: {capacity} units")  
Print("Parcels sorted by profit/weight ratio (desc):")  
For p in parcels:  
    Print(p)
```

For parcel in parcels:

```
If capacity_left == 0:
```

```
Print("Truck is full, stopping loading.")

Break

If parcel.weight <= capacity_left:
    # Take the whole parcel
    Parcels_taken.append((parcel, 1)) # 1 means whole parcel taken
    Capacity_left -= parcel.weight
    Total_profit += parcel.profit
    Print(f"\nTaking full parcel: {parcel}")
    Print(f"Remaining capacity: {capacity_left}")

Else:
    # Take fraction of the parcel
    Fraction = capacity_left / parcel.weight
    Parcels_taken.append((parcel, fraction))
    Profit_gained = parcel.profit * fraction
    Total_profit += profit_gained
    Print(f"\nTaking fraction {fraction:.4f} of parcel: {parcel}")
    Print(f"Profit gained: {profit_gained:.2f}")

    Capacity_left = 0
    Print("Truck is now full.")
```

Return total_profit, parcels_taken

```
Def main():
    Print("\nEnter number of parcels:")

    Try:
```

```
N = int(input())  
If n <= 0:  
    Print("Number of parcels must be positive.")  
    Return
```

```
Except:  
    Print("Invalid input for number of parcels.")  
    Return
```

```
Parcels = []  
For I in range(n):  
    Print(f"\nEnter profit and weight for parcel {i+1} (separated by space):")
```

```
Try:  
    Profit, weight = map(float, input().split())  
    If profit < 0 or weight <= 0:  
        Print("Profit must be >=0 and weight must be >0. Try again.")  
    Return  
    Parcels.append(Parcel(profit, weight))
```

```
Except:  
    Print("Invalid input for profit/weight.")  
    Return
```

```
Print("\nEnter truck capacity:")  
Try:  
    Capacity = float(input())  
    If capacity <= 0:  
        Print("Capacity must be greater than zero.")
```

Return

Except:

Print("Invalid input for capacity.")

Return

Total_profit, parcels_taken = fractional_knapsack(capacity, parcels)

Print("\nSummary of loading:")

For parcel, fraction in parcels_taken:

If fraction == 1:

Print(f"Took whole parcel with profit {parcel.profit} and weight {parcel.weight}")

Else:

Print(f"Took {fraction:.4f} fraction of parcel with profit {parcel.profit} and weight {parcel.weight}")

Print(f"\nMaximum Profit achieved: {total_profit:.2f}")

If __name__ == "__main__":

Main()