❖ **Knapsack problem Application:**

```
import java.util.Arrays;
import java.util.Comparator;

class Itemm {
  int weight;
  int value;
  double ratio;

  Itemm(int weight, int value) {
    this.weight = weight;
    this.value = value;
    this.ratio = (double) value / weight; // value-to-weight ratio
  }
}

public class Ecommerce {
  public static double getMaxValue(Item[] items, int capacity) {

    Arrays.sort(items, Comparator.comparingDouble((Item i) ->
i.ratio).reversed());

    double totalValue = 0.0;
    int currentWeight = 0;

    for (Item item : items) {
      if (currentWeight + item.weight <= capacity) {
        currentWeight += item.weight;
        totalValue += item.value;
      } else {

        int remaining = capacity - currentWeight;
        totalValue += item.ratio * remaining;
        break;
      }
    }

    return totalValue;
  }

  public static void main(String[] args) {

    Item[] items = {
        new Item(10, 60)
```

```
            new Item(20, 100)
            new Item(30, 120)
        };

        int capacity = 50; s
        double maxProfit = getMaxValue(items, capacity);

        System.out.println("E-Commerce Delivery Optimization");
        System.out.println("Delivery Capacity: " + capacity + " kg");
        System.out.println("Maximum Total Profit: ₹" + maxProfit);
    }
}
```

```
"C:\Program Files\Java\jdk-24\bin\java.exe"
E-Commerce Delivery Optimization
Delivery Capacity: 50 kg
Maximum Total Profit: ₹240.0

Process finished with exit code 0
```