Goods Delivery and Shortest Route Problem

Phase 1: Cargo Loading – Fractional Knapsack

Objective

- **Select goods to load into the truck (with max weight limit) to maximize total value.** You can take part of a good (fractional allowed).

Steps of the Greedy Fractional Knapsack Algorithm

1. **Calculate value-to-weight ratio** for each good:

   Ratio = $\frac{Value}{Weight}$

2. **Sort goods** in descending order by this ratio (greediest first).
3. **Iteratively add goods** to the truck:
   - Take the whole item if possible.
   - If you can't fit the whole item, take as much as fills the truck's remaining space.
4. **Stop when the truck is full.**

Example Table

Let's assume these example goods and a max truck weight of 10 kg:

Good	Value ($)	Weight (kg)	Value/Weight
1	50	5	**10.0**
2	60	10	6.0
3	140	20	7.0

Compute ratios and sort: Good 1 (10), Good 3 (7), Good 2 (6)

Loading Steps Table

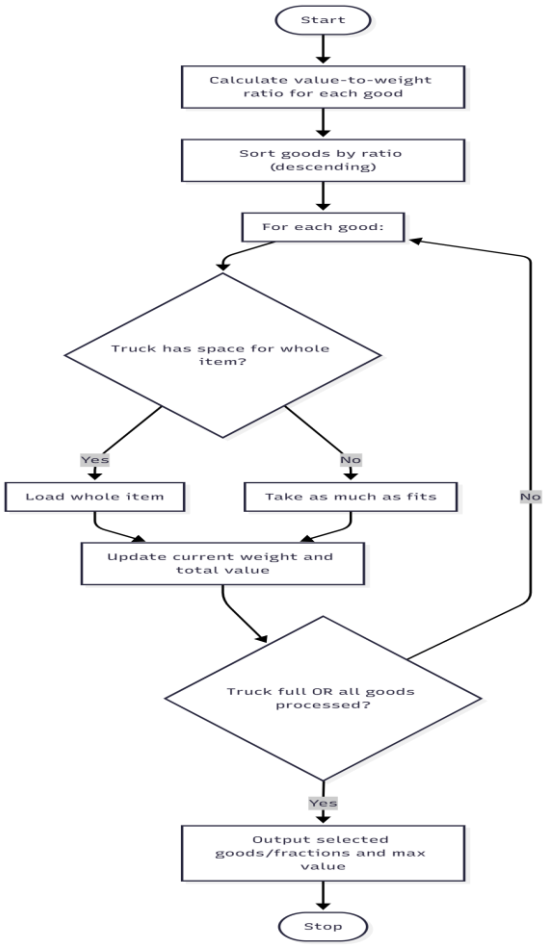
Step	Good Added	Amount Taken	Value Added	Cumulative Weight	Cumulative Value
1	Good 1	Full (5 kg)	50	5	50
2	Good 3	5 kg (of 20)	35	10	85

We ONLY have 5 kg space left after the first item. So, we take 5/20 of Good 3.

Final answers:

- Truck loaded: Good 1 (full), Good 3 (fractional)
- **Total Value:** 50+35=85
- **Total Weight:** 10 kg

Fractional Knapsack Flowchart



Phase 2: Route Planning – Traveling Salesman Problem (TSP)

Objective

- **Minimize total delivery route cost.**
- Visit each city (A, B, C, D, E) *once* and return to the starting city (full loop).

Explanation

For 5 cities: You need to find the order in which to visit all 5 so that the *total travel cost/distance* is as low as possible. This isn't "just go shortest to next"—you have to consider the complete tour, because the shortest edge-by-edge path might not be the overall cheapest tour.

Example Cost Matrix

Suppose this is your travel cost matrix:

	A	**B**	**C**	**D**	**E**
A	0	10	15	20	25
B	10	0	35	25	17
C	15	35	0	30	28
D	20	25	30	0	22
E	25	17	28	22	0

How to Find the Shortest Tour (without Coding)

- *Total unique tours for 5 cities:* (5−1)!/2=12(5-1)!/2 = 12(5−1)!/2=12 (if you treat A-B-C-D-E as same as E-D-C-B-A, ignore starting city for this math).
- List or imagine all possible city orderings, add up the cost for each full loop, and pick the minimum.

For example:

1. **A-B-E-D-C-A**
    - A→B: 10
    - B→E: 17
    - E→D: 22
    - D→C: 30
    - C→A: 15

- **Total:** 10+17+22+30+15 = 94

2. **A-C-E-B-D-A**
   - A→C: 15
   - C→E: 28
   - E→B: 17
   - B→D: 25
   - D→A: 20
   - **Total:** 15+28+17+25+20 = 105

Keep going for all (n-1)! = (5 - 1)! = 24 permutations.

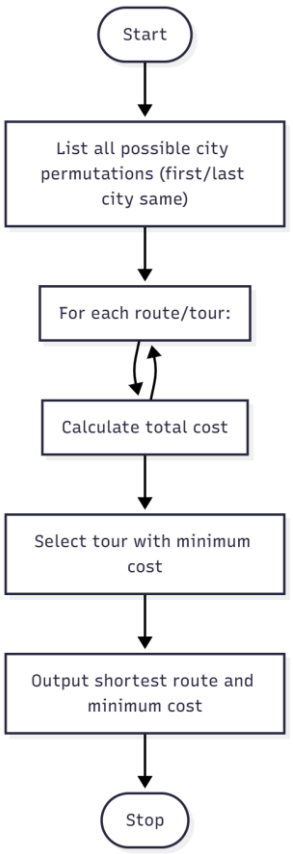
How Do You Know Which Is Shortest?

- The shortest tour will have the lowest total sum.
- This process “eliminates” all longer tours by *comparing total routes*. You're literally proving "no other tour is shorter" by checking everything.

Let’s say after trying all valid orders, you find:

- **Shortest Tour:** A-B-E-D-C-A
- **Minimum Cost:** 94

High-Level TSP Flowchart



To Summarize

- **Fractional Knapsack:** Greedily load items by value/weight ratio until the truck is full. Plug in your good's list to test yourself.
- **TSP:** List every delivery route, add their costs, and pick the absolute shortest. The brute-force way is guaranteed for 5 cities.