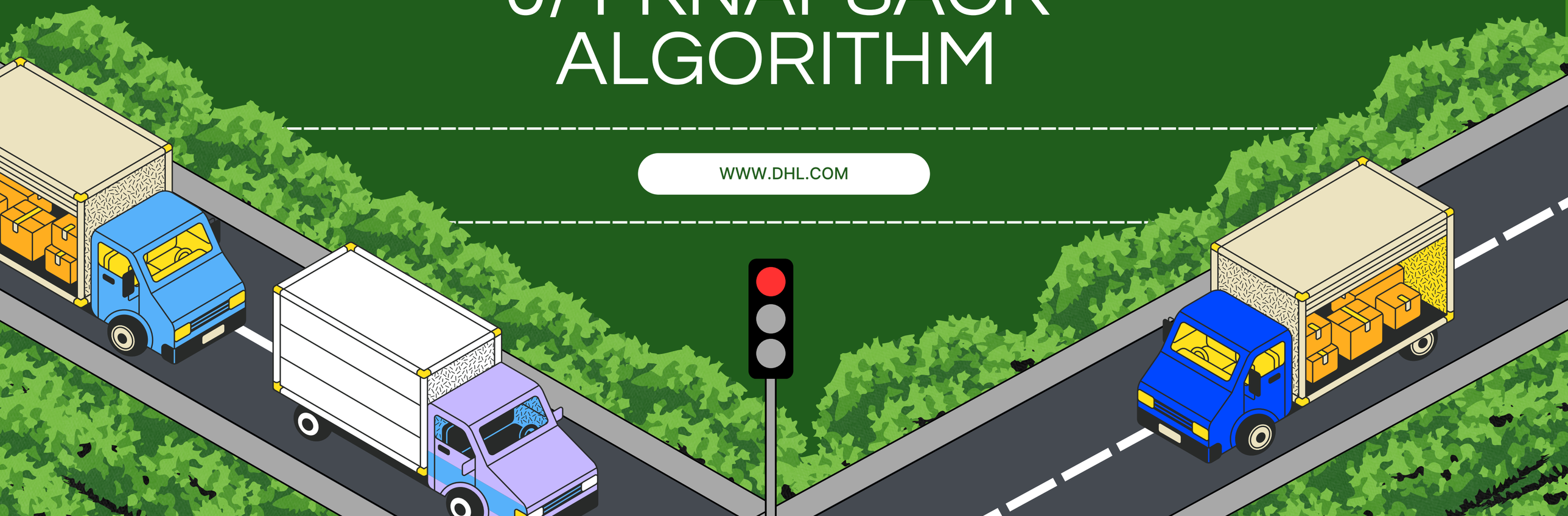
DHL LOGISTICS

OPTIMIZING CARGO LOADING USING THE 0/1 KNAPSACK ALGORITHM

WWW.DHL.COM

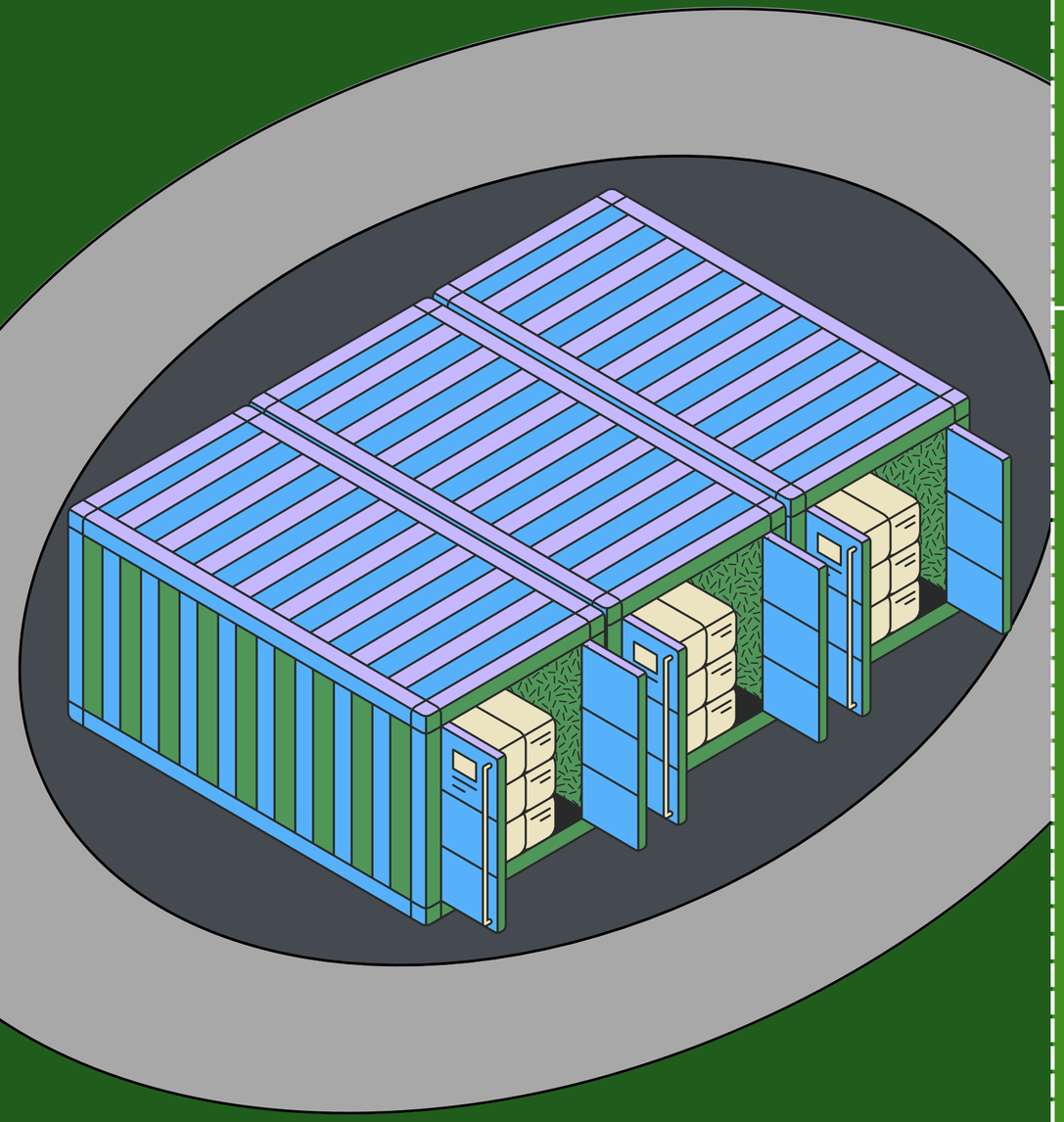


SCENARIO

DHL LOGISTICS

Handles various **logistics services** from express parcel delivery to large-scale freight movement across land, sea, and air

Reputation for **speed** and **reliability**



PROBLEMS

DHL LOGISTICS

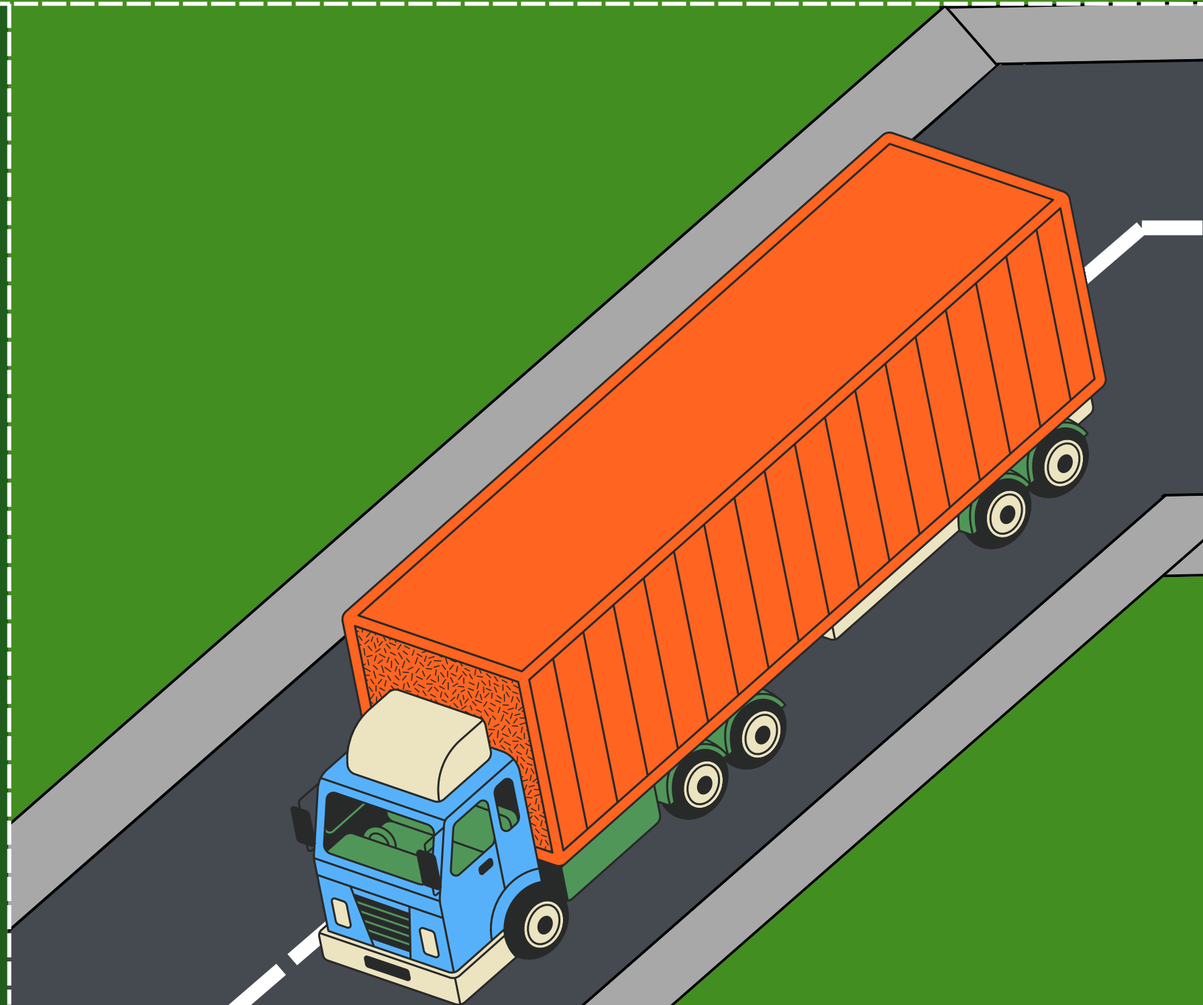
Internal reviews have shown that trucks are often **underloaded** or **inefficiently packed**

Shipments containing high-priority or time-sensitive items may be **delayed** if they are grouped with lower-priority goods in an inefficient manner

CHALLENGES IN LOGISTICS

Unhappy Customer

Increase Cost



ALGORITHM SUITABILITY REVIEW

Sorting

- Simple and fast
- Quick ranking by **value-to-weight ratio**

- **Cannot guarantee** optimal selection
- Ignore better combinations of items

ALGORITHM SUITABILITY REVIEW

Divide and Conquer

- Allows partial optimization in segmented contexts
- Scales well for distributed cargo management systems

- May not find globally optimal cargo sets unless carefully merged
- Not inherently designed to optimize cumulative cargo value

ALGORITHM SUITABILITY REVIEW

Dynamic Programming (DP)

- Provides an optimal solution using the **0/1 Knapsack Problem**
- Evaluates all viable combinations of cargo within the weight limit

- Computationally intensive for large item sets
- Requires careful modeling of item weights and values

ALGORITHM SUITABILITY REVIEW

Greedy Algorithms

- Very fast and easy to implement
- Useful for **quick approximations** or when time/resources are limited

- Does **not guarantee** optimal cargo value
- Cannot revise earlier decisions, leading to suboptimal total loads

ALGORITHM SUITABILITY REVIEW

Graph Algorithms

- Can model complex dependencies or constraints
- Useful for visualizing cargo groupings or routing with weight limits

- **Overly complex** for simple weight-based value maximization

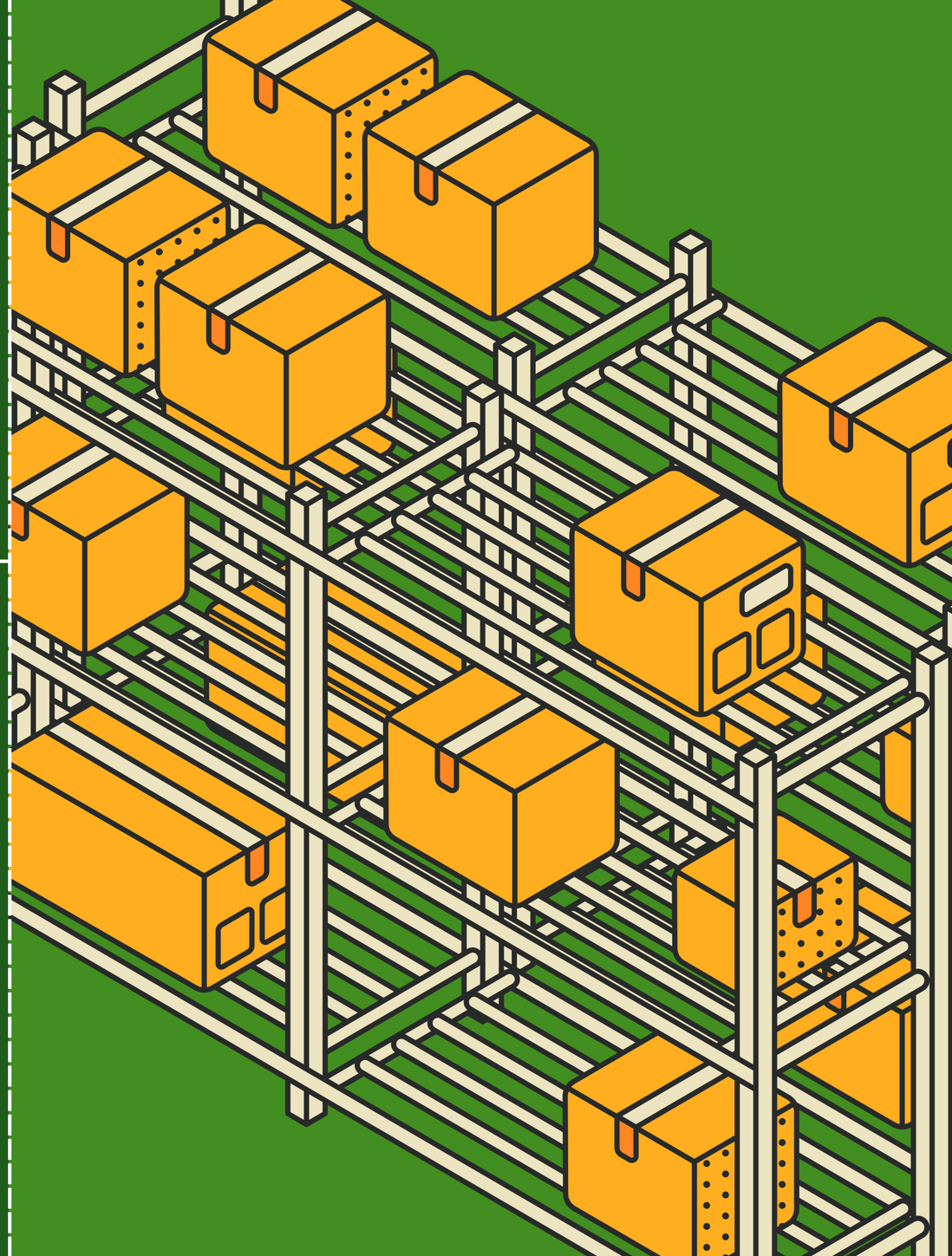
THE OPTIMIZED CHOICE:

DYNAMIC PROGRAMMING (0/1 KNAPSACK)

Maximization of total cargo value within truck’s weight capacity

Best combination of feasible items

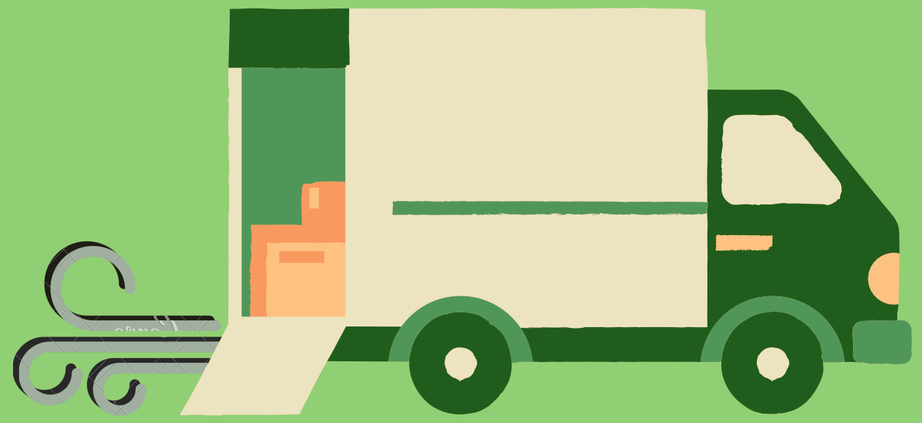
Systematic & reproducible results



ALGORITHM DESIGN

OBJECTIVE

Maximize the total value of shipments loaded onto a truck without exceeding the weight limit.



MAIN COMPONENTS

① Shipment Prioritization

- Rank shipments by value score
- Value reflects:
  - Urgency
  - Profitability
  - Delivery deadlines

② Optimal Selection via 0/1 Knapsack Problem

- Each shipment = item with:
  - Weight
  - Value
- **Goal: Maximize total value under max weight limit**

ALGORITHM APPROACH

Dynamic Programming is used to:

- Explore all possible combinations
- Track the best subset of shipments within capacity
- Ensure no shipment is partially selected (0/1 constraint)

ALGORITHM DESIGN

CARGO VALUE MODELLING

PRIORITY SCORE FORMULA

$$\text{priority\_score} = \alpha \cdot \text{revenue} + \beta \cdot \text{urgency} + \gamma \cdot \text{service\_level}$$

REASON PRIORITIZE

- Single shipment: Direct delivery
- Multiple shipments: Weight may exceed capacity → Need to select the best ones

RESULT

Priority score gives us a systematic way to balance profit, urgency, and customer tier."

COMPONENTS

Attribute	Description
Revenue	Income from shipment
Urgency	1/days remaining
Service Level	10 for Premium, 0 for Standard

DEFAULT WEIGHT

Parameter	Value
α	0.5
β	0.3
γ	0.7

EXAMPLE

SCENARIO OVERVIEW

DHL Shipment Optimization

- Goal: Maximize priority score without exceeding truck's 30kg weight capacity
- Shipment Orders: 5 incoming orders with different:
  - Weight (kg)
  - Revenue (RM)
  - Time Remaining (Days)
  - Service Level (Premium/Standard)

STEP 1: PREPARE SHIPMENT DATA & CALCULATE PRIORITY SCORE

Shipment ID	Weight (kg)	Revenue (RM)	Days Left	Service Level	Priority score
A	10	200	1	Premium (10)	107.3 ≈ 107
B	8	180	3	Standard (0)	90.1 ≈ 90
C	5	120	2	Premium (10)	67.15 ≈ 67
D	12	220	4	Standard (0)	110.075 ≈ 110
E	7	150	1	Standard (0)	75.3 ≈ 75

EXAMPLE

STEP 2: APPLY 0/1 KNAPSACK ALGORITHM

		Capacity																												
		i	0	…	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
V	W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
107	10	0	0	0	0	0	0	0	0	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107	107
90	8	0	0	0	0	0	0	90	90	107	107	107	107	107	107	107	107	197	197	197	197	197	197	197	197	197	197	197	197	197
67	5	0	0	0	67	67	67	90	90	107	107	107	157	157	174	174	174	197	197	197	197	197	264	264	264	264	264	264	264	264
110	12	0	0	0	67	67	67	90	90	107	107	110	157	157	174	174	177	197	197	200	200	217	264	264	267	267	284	284	284	307
75	7	0	0	0	67	67	75	90	90	107	107	142	157	157	174	174	182	197	197	232	232	249	264	264	272	272	284	284	292	339

STEP 3: EXTRACTING THE OPTIMAL SHIPMENT SET

- Max Priority Score: 339
- Selected Shipments (by backtracking):
  - 339 - 75(index 5's priority score) = 264
  - 264 – 67 (index 3's priority score) = 197
  - 197 - 90 (index 2's priority score) = 107
  - 107 - 107 (index 1's priority score) = 0

EXAMPLE

STEP 4: FINAL SHIPMENT SELECTION

Shipment ID	Weight	Priority Score
A	10	107
B	8	90
C	5	67
E	7	75
Total	***30***	***339***

ALGORITHM IMPLEMENTATION

DATASET

A = Shipment ID
B = Revenue
C = Time Remaining
D = Service Level
E = Weight

	A	B	C	D	E
1	1	202	2.85	10	18
2	2	370	1.13	0	65
3	3	206	4.86	0	14
4	4	171	3.93	10	8
5	5	288	4.71	0	16
6	6	120	4.5	10	18
7	7	202	3.09	0	11
8	8	221	4.63	10	56
9	9	314	0.67	0	27
10	10	187	1.18	0	35
11	11	199	0.46	10	89
12	12	251	1.8	0	5
13	13	230	2.1	10	12
14	14	249	1.54	0	5
15	15	357	4.19	10	15
16	16	393	1.94	0	65
17	17	291	1.58	0	76
18	18	376	2.83	0	78
19	19	260	0.92	10	16
20	20	121	4.06	10	10
21	21	352	0.6	0	11
22	22	335	4.94	10	9
23	23	148	3.92	0	35
24	24	158	1.19	10	5
25	25	269	0.28	0	37

	A	B	C	D	E
26	26	287	4.12	10	18
27	27	370	3.61	0	6
28	28	289	3.71	0	49
29	29	274	3.91	0	16
30	30	150	0.6	0	54
31	31	154	1.95	10	80
32	32	343	0.8	10	61
33	33	230	4.35	10	8
34	34	234	3.21	10	41
35	35	120	1.82	0	15
36	36	266	0.55	10	25
37	37	373	1.73	0	95
38	38	188	1.79	0	5
39	39	113	3.72	10	19
40	40	341	3.28	10	10
41	41	364	4.46	10	82
42	42	152	2.49	10	39
43	43	191	0.82	0	38
44	44	363	3.64	0	76
45	45	134	3.86	0	10
46	46	305	2.92	0	10
47	47	180	3.91	10	15
48	48	149	2.6	10	49
49	49	101	2.73	10	5
50	50	153	2.28	10	13

RESULT

32 selected shipment

```
Final Shipment Selection Based on Optimization:
ShipmentID   Weight (kg)   Priority Score
001          18            108
003          14            103
004          8             93
005          16            144
006          18            67
007          11            101
009          27            157
012          5             126
013          12            122
014          5             125
015          15            186
016          65            197
019          16            137
020          10            68
021          11            177
022          9             175
024          5             86
025          37            136
026          18            151
027          6             185
029          16            137
033          8             122
034          41            124
035          15            60
036          25            141
038          5             94
040          10            178
045          10            67
046          10            153
047          15            97
049          5             58
050          13            84
-------------------------------------------------
Total        499           3959
```

CORRECTNESS OF ALGORITHM

PROBLEM OVERVIEW

- Objective: Maximize total priority score of selected shipments without exceeding weight limit W = 500.
- Approach: Bottom-up Dynamic Programming.
- DP Table Definition: dp[i][w]: Maximum priority score using first i shipments with capacity w.

LOOP INVARIANT DEFINITION

- Invariant: At the start of each iteration, dp[i][w] stores the maximum total priority score using the first i shipments and capacity w.

INITIALIZATION

- All dp[0][w] = 0: No items → score = 0.
- All dp[i][0] = 0: Zero capacity → no items can be added.

CORRECTNESS OF ALGORITHM

MAINTENANCE (NESTED LOOPS)

```
for (int i = 1; i <= n; i++) {
    for (int w = 0; w <= W; w++) {
        if (weight[i-1] <= w)
            dp[i][w] = max(dp[i-1][w], priority[i-1] + dp[i-1][w - weight[i-1]]);
        else
            dp[i][w] = dp[i-1][w];
    }
}
```

- Include Case: Item fits → consider added score.
- Exclude Case: Skip item → carry forward previous best.
- Invariant Maintained: dp[i][w] always stores best score for current state.

TERMINATION & CORRECTNESS

- dp[n][W] holds the final optimal score.
- Backtracking using keep[][] reconstructs the selected shipment IDs.
- Conclusion: -Initialization, Maintenance, and Termination follow the loop invariant method.
  -The algorithm is correct and optimal.

TIME COMPLEXITY ANALYSIS

PRIORITY SCORE CALCULATION

```
for (int i = 0; i < n; i++) {
    // Compute priority score
}
```

The time complexity for this part is O(n).

DYNAMIC PROGRAMMING TABLE CONSTRUCTION

```
for (int i = 1; i <= n; i++) {
    for (int w = 0; w <= W; w++) {
        // Fill DP table using recurrence
    }
}
```

The time complexity for filling the table is O(n × W)

TIME COMPLEXITY ANALYSIS

BACKTRACKING TO FIND SELECTED SHIPMENTS

```
for (int i = n; i > 0; i--) {
    if (dp[i][w] != dp[i - 1][w]) {
        // Track selected shipments
    }
}
```

The time complexity for this part is O(n).

OUTPUT OF SELECTED SHIPMENTS

```
for (Shipment s : selected) {
    // Print details
}
```

The time complexity for filling the table is O(n × W)

BEST CASE

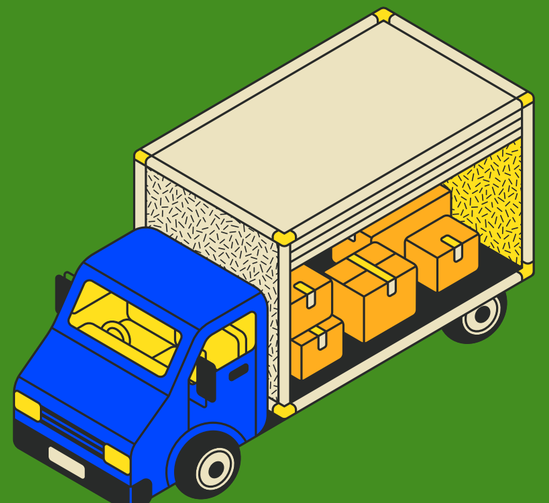
- Time Complexity: O(nW)
- The DP table of size (n+1) × (W+1) is always fully computed.
- Even if all shipments are too heavy to be included, every cell in the table is processed.
- The algorithm does not use early termination or shortcuts.

WORST CASE

- Time Complexity: O(nW)
- Every cell in the DP table is processed.
- For each (i, w), a constant-time operation is performed.
- No input configuration can avoid full-table computation.

AVERAGE CASE

- Time Complexity: O(nW)
- Some shipments are selected, others are not.
- Still, all n × W combinations of items and weights are considered.
- Variations in results do not affect the number of operations.



DHL LOGISTICS

THANKYOU

WWW.DHL.COM