Lab Project

	Marks		Project No	1
CO	**Allocated [30]**	**Obtained**	**Course Code**	CSE305L/CSE0613222L
CLO1	10		**Course Title**	Algorithm Design and Analysis Lab
CLO2	5		**Semester**	Summer 2025
CLO3	15		**Course Instructor**	Dr. Md Sajjatul Islam
			Due Date	17.08.2025
Total			**Submission Date**	
	Student ID		**Student Name**	

Bloom's Cognitive Level: C1 = Remember; C2 = Understand; C3 = Apply; C4 = Analyze; C5 = Evaluate; C6 = Create.

Complex Engineering Problem:

P1: Depth of knowledge required

P2: Range of conflicting requirements

Complex Engineering Activities:
A1: Range of resources

Project Title: Delivery Route Optimizer for Urban Logistics

Problem Description:
You are simulating a delivery routing system for a logistics company.
Given:

- A **graph-based map** (cities and roads with distances)
- A list of **delivery requests** (locations, priorities, time windows)

You must:

- Use **Dijkstra's algorithm** to compute shortest paths from the warehouse
- Use a **Greedy approach** to choose the next delivery
- Output an optimized delivery schedule.

Objectives:

- Develop a route optimizer in **C** using **graph data structures** and **Dijkstra's algorithm**.
- Apply a **Greedy heuristic** for delivery prioritization.
- Simulate conflicting requirements (shortest path vs. priority vs. time window).
- Practice **modular C programming**, **file I/O**, and **dynamic memory management**.

Complex Engineering Problem Alignment:
P1: Depth of Knowledge Required

- Graph Theory (nodes/edges, adjacency)
- Dynamic memory, pointers in C
- Algorithmic understanding of Dijkstra and Greedy

P2: Range of Conflicting Requirements

- Time window vs. shortest path vs. priority
- Vehicle capacity (optional)
- Delivery deadlines vs. efficiency

Complex Engineering Activities:
A1: Range of Resources

- File I/O in C (for input graph and delivery data)
- Memory management (malloc/free)
- Command line parameters or stdin/stdout

Expected Functional Requirements:
Data Structures:

- **Graph:** Adjacency list or matrix.
- **Priority Queue:** For Dijkstra (can use min-heap or simple linear array).

- **Delivery Info:** Struct with location, priority, and time window.

Algorithms:

- **Dijkstra's Algorithm** for single-source shortest path.
- **Greedy Heuristic**: choose next delivery based on:
  - Highest priority
  - Shortest path from current location
  - Earliest delivery window

Input/Output Specification:

Input Format (Text Files):

1. map.txt – Weighted graph

A B 4
A C 2
B D 5
C D 1
C E 7
D E 3

2. deliveries.txt – Delivery list

D High 9 10
E Medium 10 11
B Low 11 12

Expected Output (Console or File):

Starting from Warehouse: A
Delivery Sequence:
1. D (High Priority) via path A -> C -> D [Cost: 3]
2. E (Medium Priority) via path D -> E [Cost: 3]
3. B (Low Priority) via path E -> C -> A -> B [Cost: 13]

Total Delivery Cost: 19

Suggested File Structure:

/SDRO
├── main.c
├── graph.c / graph.h
├── dijkstra.c / dijkstra.h
├── delivery.c / delivery.h
├── input.txt / map.txt / deliveries.txt
└── Makefile

Evaluation Criteria:

- Correct Dijkstra implementation with adjacency list
- Greedy delivery selection algorithm
- Input/output file handling
- Memory leak-free code
- Complexity analysis of Dijkstra

Final Deliverables:

1. Source Code (C files + header files)
2. Input files (map.txt, deliveries.txt)
3. PDF Report including:
    - Problem overview
    - Objectives
    - Algorithm design and flowchart
    - C code structure
    - Example input/output
    - Mapping to P1, P2, A1
    - Time complexity analysis
    - Conclusion

1.	**Develop** a real-world solution by addressing all the conditions and objectives. (CLO1-C3) [Assessment: **Performance**, **Demonstration**]	10
2.	**Analyze the** solution. (CLO2-C4) [Assessment: **Demonstration and Project Report]**	5
3.	**Prepare** a comprehensive report covering all the components in deliverables**.** (CLO3-C2, C3, C4) [Assessment: **Project Report**]	15