	American International University – Bangladesh (AIUB) Faculty of Science and Technology (FST) Department of Computer Science (CS)	**Course: Data Structure Lab** **Final Term Assignment** **Spring 24-25**

Instructions

Use **proper C++ concepts** mentioned in **Function and System Requirements** section in the problem scenario. Use proper coding structure that should highlight your coding knowledge.

What You Will Submit

You have to create a .cpp file and the file name should be your ID. **(Example: 11-12345-2.cpp)**

Where To Submit

Submit only the .cpp file **(no zip file required)** in the assignment section named **(DS Lab Final Term Assignment Submission)** on MS Teams. After submitting the file, you must press the **"Turn In"/ "Hand In"** button.

Submission Deadline: 11th June, 2025 at 11:59 PM

What You Should Not Do

To maintain academic integrity and ensure fair evaluation, students are strictly advised against any form of plagiarism. This includes copying code or solutions from online sources such as ChatGPT, other AI tools, coding websites, or from friends. Assignments must reflect your own understanding, logic, and effort. You are also advised to follow the deadline strictly. Any violation of these rules will result in deducting marks.

Airline Baggage & Passenger Check-in System

Problem Scenario

You have been appointed to develop a system that simulates a real-time airport check-in and boarding process. This includes managing passenger data, baggage check-ins, frequent flyer records, terminal connections, and calculating overweight baggage fees. Your system must use appropriate data structures to support these operations efficiently.

Function and System Requirements

You must design and implement the following features using the data structures listed below:

1. Passenger & Baggage Entry (Doubly Linked List)

Use a Doubly Linked List to maintain a list of passengers with their baggage weights where each node should contain: Passenger Name, Passport Number, and Baggage Weight

Operations supported

- Add new passengers to the list (at the end).
- Remove passengers (from anywhere).
- Traverse forward or backward to display passenger records.

2. Check-in Correction (Stack)

Simulate an Undo feature using a Stack to reverse the last check-in entry where, each time a passenger is added or a baggage weight is updated, push that updated state onto the stack.

If a mistake is found, the system should support:

- Popping the last entry and undoing the action.
- Displaying the updated list of passengers after undo.

4. Frequent Flyer Records (Binary Search Tree - BST)

Track frequent flyers using a Binary Search Tree for efficient lookup.

Each node should contain

- Flyer ID
- Passenger Name
- Total Flights Taken

Operations supported

- Insert new frequent flyer records.
- Display flyers using in-order traversal (sorted by ID).

5. Terminal Connection Map (Graph – Adjacency List)

Represent terminals and their connectivity using a Graph (Adjacency List), where:

- Each node = Terminal;
- Each edge = Connection between terminals.

The system should support

- Adding terminals.
- Connecting two terminals with a path.
- Displaying the full adjacency list.
- **Note:** No BFS/DFS path search required.

6. Overweight Fee Calculation (Infix to Prefix Conversion + Evaluation)

Calculate final baggage charges using an infix expression such as:

- ((Weight * 100) + 200) − Discount

System should Implement

- Convert the infix to prefix.
- Evaluate the prefix using user-provided values for variables like Weight and Discount.
- Display the converted prefix and final charge amount.

Sample Input Commands in the Main Function

AddPassenger("Alice", "P12345", 18)

AddPassenger("Bob", "P67890", 30)

DisplayPassengersForward()

UndoLastCheckin()

DisplayPassengersForward()

InsertFlyer(101, "Alice", 12)

InsertFlyer(105, "Bob", 8)

InOrderFlyers()

AddTerminal("T1")

AddTerminal("T2")

AddTerminal("T3")

ConnectTerminals("T1", "T2")

ConnectTerminals("T2", "T3")

DisplayTerminalConnections()

CalculateCharge("((Weight * 100) + 200) - Discount", Weight=25, Discount=150)

Sample Outputs based on the Input Commands

Passenger Added: Alice

Passport: P12345

Baggage: 18kg

Passenger Added: Bob

Passport: P67890

Baggage: 30kg

Passenger Linked List (Forward)

(Alice | P12345 | 18kg) → (Bob | P67890 | 30kg)

Undo Last Check-in

Removed: Bob | Passport: P67890 | Baggage: 30kg

Passenger linked List (After Undo)

(Alice | P12345 | 18kg)

Frequent Flyer Inserted: ID 101 | Name: Alice | Flights: 12

Frequent Flyer Inserted: ID 105 | Name: Bob | Flights: 8

In-order Traversal of Frequent Flyers:

ID: 101 | Alice | Flights: 12

ID: 105 | Bob | Flights: 8

Terminal Added: T1

Terminal Added: T2

Terminal Added: T3

Connection Added: T1 **<->** T2

Connection Added: T2 **<->** T3

Terminal Connections (Adjacency List):

T1 → T2

T2 → T1, T3

T3 → T2

Charge Expression: ((Weight * 100) + 200) - Discount

Prefix Expression: - + * Weight 100 200 Discount

Values: Weight = 25, Discount = 150

Final Charge: 2800