CPT113/CPT113W
Programming Methodology & Data Structures
Assignment 2

Flight Arrival Management System (FAMS)

Modern airports manage dozens of flight arrivals every hour. These arrivals must be processed, prioritised, queued, and recorded accurately to ensure safe and efficient airport operations. Poor handling of arrival information can lead to congestion, delays, and miscommunication between airport units.

In this assignment, you are required to design and implement a **Flight Arrival Management System (FAMS)** using Object-Oriented Programming (OOP) concepts in C++, strictly following the data structure implementation approach taught in the lectures and based on the D.S. Malik textbook. The system will read flight arrival information from a text file once only, then organise and manage the data using three fundamental Abstract Data Types (ADT):

- Sorted Linked List – to maintain all incoming flights in ascending order of arrival time
- Linked Queue – to manage flights waiting to land
- Linked Stack – to store flights that have already landed

Flight Arrival Data

Each flight arrival record in the input text file contains the following fields:

- Flight Number (string)
- Departure Airport Code (string)
- Arrival Airport Code (string)
- Arrival Time (24-hour format: HHMM)
- Flight Type (DOMESTIC / INTERNATIONAL)
- ETA Status (ONTIME / DELAYED)

Example input text data

MH101 KUL PEN 0915 DOMESTIC ONTIME

AK203 JHB KUL 0845 DOMESTIC DELAYED

SQ305 SIN KUL 1030 INTERNATIONAL ONTIME

MH220 PEN KUL 0900 DOMESTIC ONTIME

PROGRAM DESIGN REQUIREMENTS:

Your solution **must strictly follow the OOP and data structure approach taught in this course**

- Create a **`class Flight`** to store appropriate data as a private data members and methods include constructors, accessor (getter) methods, mutator (setter) methods and a display method.
- Read flight arrival data from the text file. Use the appropriate data structure to sort incoming flights, put them in the landing queue, and store the most recent landing flight.
  [Note: The text file should only be read once, and then the relevant data should be stored in appropriate data structures]
- **The implementation of the program must meet the following specification requirements.**
  - Must use object-oriented approach taught in the course
  - Do not use STL containers such as vector, list, queue, or stack
  - Implement separate classes for the following ADTs template style:
    - linkedListType (Sorted Linked List)
    - linkedQueueType (Queue)
    - linkedStackType (Stack)
  - Store an object of the class into appropriate data structure
  - Global variables are **NOT** allowed in your program.
  - **User defined functions** are **NOT** allowed except for reading text data and menu functions

Functional Behaviour Details

- **Read Flight Arrival Data**
  - Reads the text file **only once**
  - Stores all records into the sorted linked list
- **Landing Process (Simulation Requirement)**
  - Each time a flight is deleted from the landing queue, it is considered **landed**
  - The landed flight must be pushed into the stack
- **Search Functions**
  - Allow users to search flights by flight number
  - Clearly indicate if a flight is not found
- **Statistics**
  - Must be computed dynamically using the stored data structures

Menu and Functionality Specification

The Flight Arrival Management System shall provide the following menu to allow users to interact with the system. The menu must repeatedly appear until the user chooses to exit.

Flight Arrival Management System (FAMS)

---

1. Read Flight Arrival Data
2. Display All Arriving Flights (Sorted by Time)
3. Display Landing Queue
4. **Process Flight Landing**
5. Display Landed Flights (Most Recent First)
6. Display Delayed Flights Only
7. Search Arrival Information by Flight Number
8. Display Arrival Statistics
9. Exit

Enter your choice:

Note: Menu no. 4 Process Flight Landing:

Flight landing must be processed explicitly via a menu option no. 4. Each selection of the landing operation will delete exactly one flight from the landing queue and push it onto the landed flights stack

General Requirements

- Input validation for menu choices and user input
- Clear, user-friendly menu and output formatting
- Meaningful variable names and comments
- Proper indentation and coding style

SUBMITION INSTRUCTIONS:

1. ZIP all the related source code and name the zipped file as follow:

   Assign2_Student1_matric No._Student2_Matric No.

   Example: `Assign2_12345_22334.zip`

2. Upload the Zipped file to eLearning
3. **Deadline: 11 January 2026 (11.59 PM)**

Course Policy:

All assignments MUST be submitted before/on the given date. Late submissions without prior approval from the lecturer will not be accepted. 8 marks will be deducted for each day of late submission.

Plagiarism or copying are serious academic offenses. Students who are found to plagiarize or copy will receive an F for the assignment or the entire coursework grade and will be barred from taking the final examination. It is important to read your undergraduate Programme Handbook.

The use of GPT, AI tools, or external consultants to complete your assignment is strictly prohibited.