**Data Structures Assignment #1 - Fall 2023 [Flight Management System]**

**Important Rules:**

* Take attention to handle all exception cases by the user throughout the Assignment.
* You MUST understand everything in the code.
* Code quality is measured according to naming conventions, writing comments, following the dynamic and static nature of program logic.
* You’ll submit a file with members ids and with .cpp extension. ex [20000000.cpp]
* Grades are based only on the discussion.
* **Deadline: 16/11/2023**

**Assignment Requirements:**

# Part #1 [Flight Class]: 1 Grades

* implement a base class called **Flight** to represent the **trip**,
* this class contains information about the **Flight** such as **private** attributes [*Flight\_ID* (int), *Model* (string), *Capacity (*int), *Airline* (string), *Destination* (string), *Departure\_Time* (string), Ticket\_*Price* (double) ],
* You must implement a parameterized **constructor** for this class that takes the attributes from the user as parametersand then sets the attributes with the values received in the constructor, so that you can define a new **Flight**.
* Implement all required **setters**() **and** **getters**() for the attributes
* **Flight** class contains a **void** method to print all of the **Flight** information method name is **display\_Flight\_info().**

# Part #2 [Linked List Class]: 3 Grades

* You must implement a Node class to represent a ode of a linked list, it should be generic as well so it can accept any type of data, this class contains two attributes **ONLY** the data part which we will store the object of type **Flight** in it, and the pointer **Next. (Look for the code tips for more info.),** this class contains one **constructor** to take the data and set its data variable and set the next to NULL. ( **LinkedList** and **FlightStack** will use this class)
* You must implement a Singly Linked List of type *Flight* using the concept of **Generic (You will find helpful code tips on Moodle)**, with name ***FlightList***this linked list should contain the following methods:
* **AddFlight()** insert a new Flight at the end of the list,
* **DelFlight()** delete a Flight from the list by flight\_ID,
* **DisplayAll()** print all Flights in the system.
* **Insert new flight**

# Part #3 [Stack using Linked List]: 4 Grades

In this part you’re requested to implement a **Stack** using Linked List not using Arrays, you need to implement the following: (Consider that the stack is FILO or LIFO)

* Class **FlightStack** to represent the stack, it will be **template** so that it can accept any type, this class has one attribute which is the **TOP (TOP is like HEAD),** and one **constructor** to set **TOP** to NULL.
  + This class will have a method **PushFlight()** to add flight to the stack should compare the new flight data to existing flight in the system and reject duplicates.
  + This class will have a method **PrintStack()** to print the flights list in LIFO order
  + **Peek()** method to return the latest flight, which is the TOP of the stack
  + Implement a method to check if the stack is empty with name bool **IsEmpty(),** because you can’t call Peek() if the stack is empty.
  + Implement a methods that gives the user the ability to **edit** or **delete** the flight information from the stack.

# Part #4 2 Grades

* To run the program the main method should print a welcome message to the user, then a menu of options as follows:

1. LinkedList.
2. Stack

* LinkedList:

1. Insert new flight
2. Delete Flight
3. Display All Flights
4. Close the program

* Stack:
  1. Insert new flight
  2. Check the latest Added Flight
  3. Print all the flights
  4. Edit flight
  5. Delete flight
  6. Close the program
* You need to handle if the user wants to choose another option and so on.