# Class Outline

## Airport Terminal Simulation
## Programming Assignment 1

21 February 2021

**Prepared By**
Nolan Anderson
npa0002@uah.edu

**Prepared for**
Dr. Jacob Hauenstein
CS 307, Object Oriented Programming
Computer Science Department
University of Alabama in Huntsville

## 1.0 System Overview

To solve the problem of parsing XML files comprised of a large number of different flights (consists of departure time and city, destination city, aircraft type etc.) requires the creation of multiple classes. This software will largely be ran by the class outlined in section 3.0, with the following classes (section 4.0-6.0) parsing the data. In other words, sections 4.0-6.0 will provide section 3.0 with the data it needs to do its calculations and output. Mostly, sections 4.0-6.0 call the parsing functions and return data and they are much simpler than section 3.0. The output of the software is simple. It outputs all of the data of each flight every 5 seconds. This data includes the flight number, aircraft name and type, departure time and city, destination city among many other different data points. In summary, the main functionality of the program is to send out flight data. If you were the flight controller for every flight in the United States, this would provide you with every flight that is going out or currently in transit.

## 2.0 **Relevant Terms and Acronyms**

Class – This is a user-defined data type that we can use to hold member variables and functions.
Member Variables – The pieces that make up a class.
Member Functions – The "doers" of the class, as in they perform the calculations, output etc.
InFile – A standard name used for input file names.
N/a – This thing or value does not exist and is not needed in this function.
Void – A function type that does not need to return a value. Usually a function that creates, or performs output.

## 3.0 Flight Simulator Executor

### 3.1 Class FlightSim

#### 3.1.1 Member Variables

Multiplier – Variable to switch how quickly the simulation runs (1X 2X 3X speed).
InFile – .txt file name to get the data for the program to parse.
FlightNum – The number of the flight, used in other functions to get data.

#### 3.1.2 Member Functions

3.1.2.1 - FlightSim::FlightSim();
Actions Performed – Creates a new instance of the fligh simulation.
Calls 3.1.2.3, 3.1.2.4, and 3.1.2.5.
Arguments – N/a.
Return Value – N/a.

3.1.2.2 - ~FlightSim::FlightSim();
Actions Performed – Deconstructs the current instance of the simulation.
Arguments – N/a.
Return Value – N/a.

3.1.2.3 - int SetMultiplier();
Actions Performed – Sets the time multiplier for the program output.
Arguments – N/a.
Return Value – Integer of the multiplier.

3.1.2.4 - string SetInFile();
Actions Performed – Sets the name of the input file.
Arguments – N/a
Return Value – Returns a string of the name of the input file.

3.1.2.5 - void Start(int Multiplier, string InFile);
Actions Performed – The main function that starts and runs the simulation.
Arguments – The multiplier speed and InputFile name.
Return Value – Void.

3.1.2.6 - int CurrentLocation(int FlightNum, int CurrentHr, int CurrentMin);
Actions Performed – Calculated the current location of the flight.
Arguments – Flight number, and the simulations current time in hours and minutes.
Return Value – Returns an integer of the location.

3.1.2.7 - int FlightTime(int FlightNum);
Actions Performed – Returns the projected flight time.
Arguments – The flight number.
Return Value – An integer of the projected flight time.

3.1.2.8 - void OutNewFlight(int FlightNum);
Actions Performed – Outputs a new flight.
Arguments – The flight number.
Return Value – Void.

3.1.2.9 - void OutInterval(int FlightNum);
Actions Performed – Outputs all of the current flights, intervals.
Arguments – The flight number.
Return Value – Void.

There will be get and set functions defined for each private member variable.

## 4.0 Populating City Data from XML Files.

### 4.1 Class CityData

#### 4.1.1 Member Variables

CityName – The Name of the City
StateName – Name of the state the city is in.
CityLat – Latitude of the City.
CityLon – Longitude of the City.
Distance – Distance to the other City.

#### 4.1.2 Member Functions

4.1.2.1 - CityData::CityData(string InFile);
Actions Performed – Constructs a new instance of the city data.
Arguments – The name of the input file.
Return Value – N/a.

4.1.2.2 - ~CityData::CityData();
Actions Performed – Deconstructs the current instance of the City Data class.
Arguments – N/a.
Return Value – N/a.

4.1.2.3 - void SetData(string InFile);
Actions Performed – Calls the XML parser to set data for section 4.1.1.
Arguments – The name of the input file.
Return Value – Void.

4.1.2.4 - string ReturnName(int FlightNum);
Actions Performed – Returns the name of the city.
Arguments – The flight number.
Return Value – A string of the city name.

4.1.2.5 - string ReturnState(string CityName);
Actions Performed – Returns the state the city is in.
Arguments – The name of the city.
Return Value – A string of the city name.

4.1.2.6 - float ReturnLatitude(string CityName);
Actions Performed – Returns the latitude of the city.
Arguments – The cities name.
Return Value – A float of the City's Latitude.

4.1.2.7 - float ReturnLongitude(string CityName);
Actions Performed – Returns the Longitude of the City.
Arguments – The cities name.
Return Value – A float of the longitude.

4.1.2.8 - float ReturnDistance(int FlightNum);
Actions Performed – Returns the destination distance based on the flight number.
Arguments – The flight number.
Return Value – A float of the distance.

There will be get and set functions defined for each private member variable.

## 5.0 Populating Flight Data from XML files.

### 5.1 Class FlightData

#### 5.1.1 Member Variables

Airline – Name of the airline
AircraftType – The type of the aircraft.
DepartureCity – The city the FlightNum is departing from.
DestinationCity – The intended arrival city of FlightNum.
FlightNum – The number of the flight.

#### 5.1.2 Member Functions

5.1.2.1 - FlightData::FlightData(string InFile);
Actions Performed – Constructor for the flight data, calls SetData.
Arguments – Name of the input file.
Return Value – N/a.

5.1.2.2 - ~FlightData::FlightData();
Actions Performed – Deconstructor for the current instance of the FlightData class.
Arguments – N/a.
Return Value – N/a.

5.1.2.3 - void SetData(string InFile);
Actions Performed – Calls the parsing functions to populate 5.1.1 Variables.
Arguments – Name of the input file.
Return Value – Void.

5.1.2.4 - string ReturnAirline(int FlightNum);
Actions Performed – Returns airline name based on the flight number.
Arguments – The flight number.
Return Value – A string of the Airline name.

5.1.2.5 - string ReturnType(int FlightNum);
Actions Performed – Returns type of aircraft based on the flight number.
Arguments – The flight number.
Return Value – A string of the aircraft type.

5.1.2.6 - string ReturnDepCity(int FlightNum);
Actions Performed – Returns departure city based on the flight number.
Arguments – The flight number.
Return Value – A string of the name of the departure city.

5.1.2.7 - string ReturnDestCity(int FlightNum);
Actions Performed – Returns destination city based on the flight number.
Arguments – The flight number.
Return Value – A string of the destination city.

5.1.2.8 - int ReturnTime(int FlightNum);
Actions Performed – Returns time of the flight.
Arguments – The flight number.
Return Value – An integer of the departure time.

5.1.2.9 - int ReturnFlightNum();
Actions Performed – Returns the flight number.
Arguments – N/a.
Return Value – An integer of the flight number.

There will be get and set functions defined for each private member variable.

## 6.0 Parsing Aircraft Data from XML files.

### 6.1 Class AircraftData

#### 6.1.1 Member Variables

Make – The make of the aircraft.
Model – The model of the aircraft.
Speed – The flight speed of the aircraft.
ClimbSpeed – The speed at which the aircraft gets to cruising altitude.
WingSpan – The wingspan of the aircraft.
FuselageLength – The length of the fuselage.

#### 6.1.2 Member Functions

6.1.2.1 - AircraftData::AircraftData(string InFile);
Actions Performed – Constructor for the Aircraft data.
Arguments – The name of the input file in terms of a string.
Return Value – No return value, constructor.

6.1.2.2 ~AircraftData::AircraftData();
Actions Performed – Deconstructor for the aircraft data, will clear the data parsed.
Arguments – N/a.
Return Value – No return value, deconstructor.

6.1.2.3 - void SetData(string InFile);
Actions Performed – Sets the data of variables in section 6.1.1 by parsing the data in InFile.
Arguments – The name of the input file in terms of a string.
Return Value - Void

6.1.2.4 - string ReturnMake(int FlightNum);
Actions Performed – Returns the make of the aircraft based on the outgoing flight number.
Arguments – The outgoing flight number.
Return Value – The name of the make of the aircraft.

6.1.2.5 - string ReturnModel(int FlightNum);
Actions Performed – Returns the model of the aircraft based on the outgoing flight number.
Arguments – The outgoing flight number.
Return Value – The name of the model of the aircraft.

6.1.2.6 - int ReturnSpeed(string model);
Actions Performed – Returns the cruising speed of the aircraft.
Arguments – Name of the model of the aircraft (string).
Return Value – Integer of the cruising speed.

6.1.2.7 - int ReturnClimb(string model);
Actions Performed – Returns the speed at which the aircraft gains altitude.
Arguments – Name of the model of the aircraft (string).
Return Value – Integer of the climb speed.

6.1.2.8 - int ReturnWing(string model);
Actions Performed – Returns the length of the wingspan.
Arguments – Name of the model of the aircraft (string).
Return Value – Integer of the length of the wingspan.

6.1.2.9 - int ReturnFuselage(string model);
Actions Performed – Returns the length of the Fuselage.
Arguments – Name of the model of the aircraft (string).
Return Value – Integer of the length of the aircraft.

There will be get and set functions defined for each private member variable.