# 



Name: Class:

Ibrahim Ahmad SE-A

Roll Id: Date:

22i-8781 10-05-2023

**OOP Project Document**

Course Instructor: Mam Maheen Arshad

[Introduction: 3](#_Toc134652413)

[OOP Concepts Used: 4](#_Toc134652414)

[Inheritance: 4](#_Toc134652415)

[Aggregation: 5](#_Toc134652416)

[Polymorphism: 6](#_Toc134652417)

[Composition: 7](#_Toc134652418)

[Class Diagram: 8](#_Toc134652419)

[Conclusion: 8](#_Toc134652420)

# 

# Introduction:

This 2nd semester OOP course project “NUCES Airline Flight System” is a management system where we were to use various OOP concepts. The project has been done with “Visual Studio” as IDE and “Draw.io” for class diagram.

This report provides a complete explanation of the OOP concepts used in the development of the airline flight system project. It includes a detailed explanation of inheritance, aggregation, association, encapsulation, polymorphism, and composition, along with screenshots that demonstrate how these concepts were implemented in the project. Additionally, the image of the class diagram for the “NUCES Airline Flight System” is included in the report.

The report also includes a detailed description of the “NUCES Airline Flight System” project, its features, design, and implementation. The testing and evaluation process used to ensure that the project meets its objectives is also discussed in the report. The input validation and other breakthroughs. Though there are things that can be improved regarding security, input validation, and exception handling.

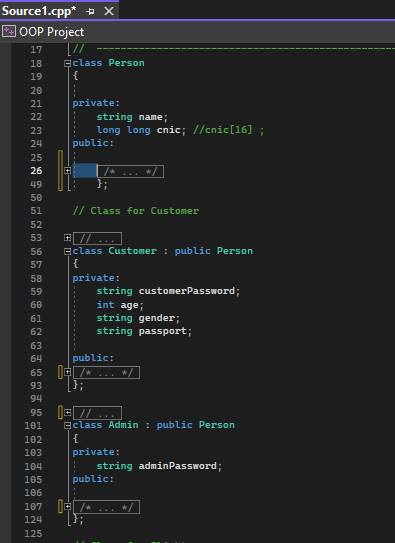
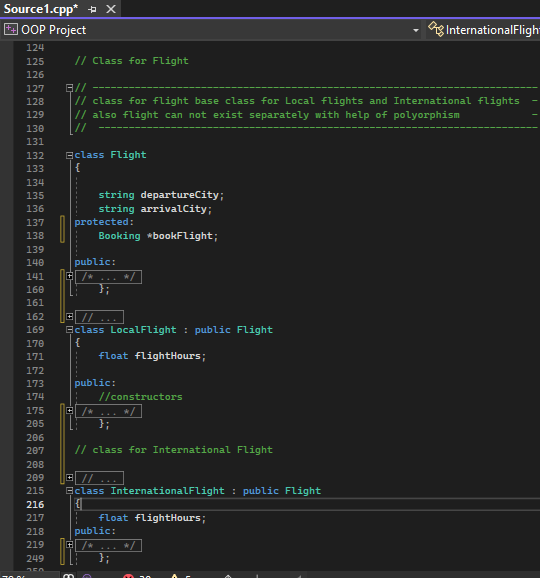
Overall, this report provides a comprehensive overview of the airline flight system project and its OOP concepts, design, and implementation. The report is organized clearly and logically, making it easy to follow and understand.

# 

# OOP Concepts Used:

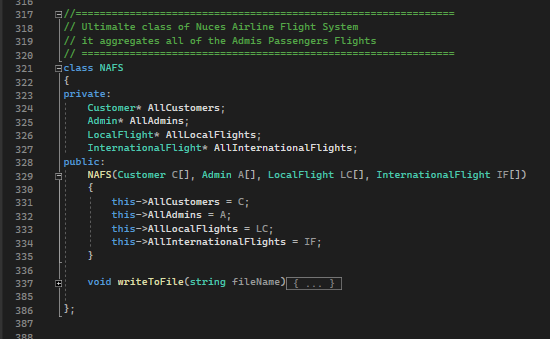
## Inheritance:

Inheritance in a basic concept of OOP increases readability and reduces code complexity. Using Inheritance, we should remember that the relation between classes should be “Is-a”. Now considering this project there was the requirement of making an “Admin” class and a “User” class but there are attributes that are common in both of these classes which are attributes of person so I made person as parents’ class and “Admin” and “User” are inherited from it. The same was the case of Flight “Local Flights” and “International Flights” so i made Flight as parent class and inherited “Local Flights” and “International Flights” from it as in the picture below.



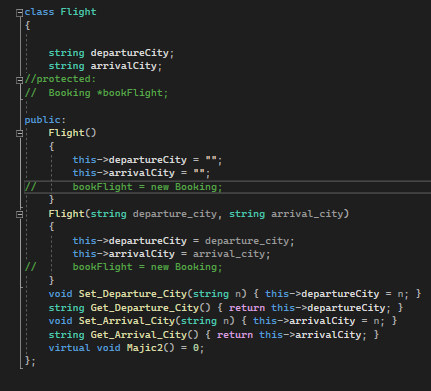
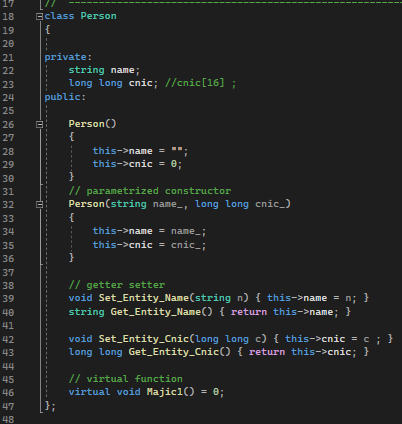
## Aggregation:

Aggregation is an important concept of OOP. Using Aggregation, we should remember that the relation between classes should be “Has-a”. Now considering this project there was the requirement of making different classes of “Admin” and a “User” and “Local Flights” and “International Flights”. We made the object of each class during the program so to store the whole data in one management system big class we use aggregation. The part classes in aggregation can exist without the whole class so we made one large class NAFS which aggregates all class objects array which can store all data as in the picture below.



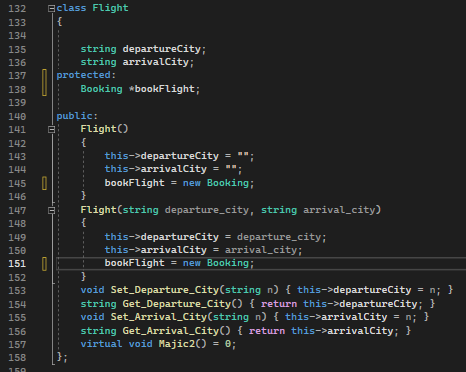
## Polymorphism:

Polymorphism is a very interesting concept where you use virtual words and can make a class abstract so its object cannot be used. This project, helped me in “person” and “Flight” where no person and flight object should exist because they were basic attributes common in different classes so I used a void function called the “Majic” function to make these classes abstract also you must remember to override the virtual function in inherited classes otherwise they would also become abstract. As in pictures.

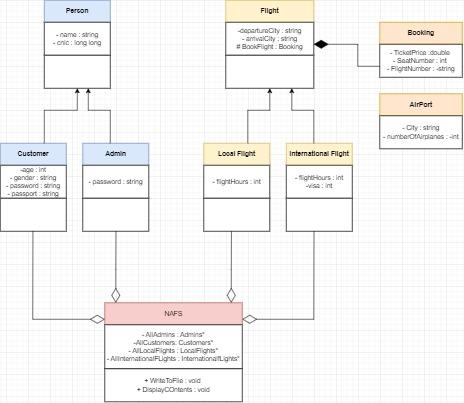


## Composition:

Composition is an important concept in OOP. considering a Part-Qhole relationship in the composition of classes, classes have a relation of “Part-of”. Now in this project, I made a booking class which was composed in flight class so the booking creation and destruction is by flight. If the flight is somehow canceled the Booking is all canceled. And no more bookings exist without Flights. Flight is an abstract class but as Booking is common in both local and international flights so it will inherit both classes.



# Class Diagram:

The class diagram is a visual way of representing code. Making class diagrams before coding helped me a lot and saved my time as it cleared my mind that what and how i have to make things work. This class diagram is build using “Draw.io” as UML.

# Conclusion:

This project was a very interesting way to help us string our concepts of OOP and use it in our daily life applications. And in the end, I want to Thanks my teachers who put in their full effort to help us understand these difficult subjects this easily.