Airport Management System

Submitted By:

Srashti Soni

Under the Guidance of Prof. Donghwa Kim

Data 690: Data Structures and Algorithms in Python

Introduction

- Airport Management System is designed to managing flights, passengers, and airports. The system provides several features like adding flights, passengers, and airports, displaying details of flights, passengers, and airports, searching for flights, sorting flights, and displaying flight details.

- The system is implemented using object-oriented programming principles, which makes the code more modular and easier to maintain.

System Design

Implementation of 5 classes: Person, Passenger, Employee, flight and Airport.

Passenger and Employee classes are the child class of the Person class.

Flight class contains several methods like add_passenger, remove_passenger and get_passenger_by_name.

Airport class contains methods like add_flight, remove_flight, get _flights, search_flight_by_numbers and sort_flights method to sort the flights according to the departure time.

Concepts used: Inheritance and Polymorphism

- Inheritance is a mechanism in object-oriented programming (OOP) that allows one class to inherit properties (fields and methods) from another class. In other words, the new class (subclass) is based on an existing class (superclass), and it can reuse and extend the behavior of the superclass.
- For example, in this project Person class is the base class and Passenger and Employee classes are the subclasses extending the properties of its super class.
- Polymorphism refers to the ability of objects of different classes to be used interchangeably in a program. Polymorphism allows objects of different classes to be treated as if they were objects of the same class, which can help reduce code complexity and increase flexibility.

Computational Complexities of Searching and Sorting Algorithm

Binary Search: The time complexity of binary search is O(log n), where n is the number of elements in the list. This is because in each iteration, the size of the search range is halved. As a result, binary search can be much faster than linear search for large datasets. Binary search has a space complexity of O(1), as it does not require any additional memory beyond the input array.

Merge Sort: The time complexity of merge sort is O(n log n), where n is the number of elements in the list. This is because the algorithm divides the list into halves recursively, sorts each half, and then merges the sorted halves. Merge Sort has a space complexity of O(n), as it requires additional memory to store the sorted subarrays during the merging process.

Features:

01

Creating an airport.

02

Adding flights to the airport.

03

Adding and removing passengers in the flight.

04

Getting list of all the flights.

05

Searching a flight from the list of all the flights.

06

Sorting the flights on the basis of departure time.

References:

- Kalb, I. (2022). Object-Oriented Python: Master OOP by Building Games and GUIs. No Starch Press.
- Estivill-Castro, V., & Wood, D. (1992, December). A survey of adaptive sorting algorithms. ACM Computing Surveys, 24(4), 441–476. https://doi.org/10.1145/146370.146381
- Akl, S. G., & Rheinboldt, W. (2014). Parallel Sorting Algorithms. Elsevier Gezondheidszorg.