Jack Bergin and Jered Kalombo
Data Structures and Algorithms
Professor Mansouri
Sep 15, 2024

Complexity Order and Runtime

5401 milliseconds to read all file details. Complexity Order: O(n)

-This loop goes through each flight in the flight array once and moves on to the next. If there are n flights, then the loop runs n times. Therefore, the large runtime of 5401 ms only shows it processing each flight as the time increases linearly with the number of flights.

83 milliseconds for task 4.a Complexity Order: O(n)

-This algorithm iterates over each flight to check the origin or destination for a Maine location. The time for the operation is constant for each flight. Since each flight is only checked once, the algorithm runs in a linear time, O(n), where n is the total number of flights.

66 milliseconds for task 4.b Complexity Order: O(n)

-This algorithm also iterates over each flight, yet it compares the number of passengers only if the destination is PWM. The runtime of 66 ms is quite low, which is expected as the algorithm updates on a linear scan of the flights.

104 milliseconds for task 4.c Complexity Order: O(n)

- This algorithm also begins with the same for loop, therefore iterating over each flight, checking for flights with no empty seats. This is, again, an operation where the time is constant for each additional update. The runtime of 104 may be a bit larger due to the number of comparisons between each flight.

65 milliseconds for task 4.d

- Following the previous four algorithms, this algorithm begins by iterating over each flight and searches for multiple conditions: origin, destination, and year. Each check has a constant time, and the loop is executed n times. The runtime of 65 ms is low, as expected for a linear algorithm.