Program Structure by Sasha Bakker

Variables/Structures

- `dict` struct holds data for each individual flight
- `departTimes` array holds all times with ATL departures
- `arriveTimes` array holds all times with ATL arrivals
- `numDeparts` integer is the number of ATL departures
- `numArrivals` integer is the number of ATL arrivals

Functions

- `printValues()` prints flight data
- `readFile()` stores all flight data in a `dict`
- `swap()` swaps values of two variables
- `heapify_up()` and `heapify_down()` used in `heapSort()`
- `merge()` used in `mergeSort()`
- `insertSort()`
- `ComputeMaxAndMin()` finds the maximum and minimum numbers of flights in the day and computes/prints the number of gates and planes needed by midnight

Main()

// create dict

// readFile() to put values in dict

// initialize departTimes, arriveTimes, numDeparts, numArrivals

// Store departure and arrival time data

// Sort arriveTimes and departTimes via insertSort(), heapSort(), or mergeSort()

// Use sorted departure and arrival time data in ComputeMaxAndMin()

Program Performance

The average/worst case time complexity for insertion sort is n^2 and best case is n. The average/worst case time complexity for both merge sort and heap sort is nlogn. Thus, we would expect these sorts to have similar run time. When the program executes, it is so fast that it is indistinguishable as to which performs the fastest.

I believe the program works correctly because:

1. When printing out the sorted data of combined arrival and departure times in ComputeMaxAndMin(), the times are sorted in increasing order and arrivals always occur before departures.
2. The output of each of the sorting methods produces the same result

Peer Review Process

- The peer review process was useful because the suggestions from other students helped me fix my calculation for gates which could be re-used
- The peer review process was not useful for this particular assignment because there is not much to improve once the sort algorithm works except for comments/how output is printed/how output is calculated after the sort. Viewing other students' code wasn't super useful because there aren't many different ways the sorting methods can be written and other students seemed just as confused about the final calculation for gates/planes by midnight