Airport Database Document

**Name: Michael Coleman**

**GMIT Number: G00347650**

The first option the user must enter is the username and password, I decided to use structs for holding the login details to keep them organized together. The user is given an option to enter the username and password, which must match that in login.txt. each user has a set password which must match each other. (E.g. user1 must match with passw1 and not passw2 / 3 etc.) The password is protected for security as \*’s are placed on the screen, this is achieved using getch() and putch() functions to hide the input and replace with \*’s. Program also closes to ensure security if wrong password is entered and will automatically close if user enters too long of a password.

**Usernames(Passwords):** user1(passw1), user2(passw2), user3(passw3)

Passenger details are loaded into the linked list from passengers.txt file, by running a for loop the length of the linked list which was saved into the txt file. newnode memory is allocated each loop, data then is passed into newnode and after the 1st node is setup, curr is set to \*top for allocation of nodes to follow the 1st. (As 1st node must be setup differently to the ones which follow). The nodes are then sorted using their passport numbers to sort them in order

Passengers can then be added to the linked list and is setup by whether the linked list already contains a node (i.e if not will run addPassengerToListStart () function or else its added to the end. ). User must also ensure that there are no duplicate passport numbers when adding and that email addresses contain an @ and .com. Passengers are then sorted based on their passport number, this is done using nested for loops to store and overwrite using a temp variable if the passport numbers are not in the correct order. (so, passport number can be deleted and reused again when adding and will be placed in correct position)

Note: Travel classes shortened input for user when adding node e.g.

Economy- (stays as is)

Premium Economy - Premium

Business Class - Business

First Class – Firstclass

**Also**(for area travelled from) Rest of Europe - RestofEurope

Display functions such as display all passengers and a specified passenger are available. The passenger’s details can also be edited via the passport number, I chose the passport number as this was assured to be a unique number, whereas the name I cannot be sure whether 2 passengers might happen to have the same name by chance. While searching for the passport number once encountered, details are requested to be changed and a break occurs to exit search, while if it is not found user is told passport number doesn’t exist.

Delete functions also available for start and end as well as deleting by passport number, if passport number is greater than the length list, then last list item removed, while if a passport number within list range isn’t there, user is notified (i.e. user deletes pass number 14, then tries to delete again, they are notified it was not fond.)

User can select statistic based on travel class. I chose to then give the user the option to select which travel class they would like to view stats on, based on their choice rather than all 4 outputting at once, data is sent through to another function where the percent for their chosen class is calculated and displayed. I chose to divide this area up into more than one function as it required a lot of code for each class, as well as code of passengers born before 1980. As just mentioned user also have same choice to choose stats based on passengers born before 1980. The stats are calculated by comparing string case insensitive(stricmp()) to that of area stored in the linked list and accounted for, the average days is calculated by logical steps using <,> and = using the duration stored in linked list.

Passengers details can also be printed to the passenger file by using the menu option or by exiting the application. The length of the list is also outputted as it is needed for reloading back into the linked list.

For sorting the list by UK passengers in order of their D.O.B. I create a new linked list and copy over the exact linked list to avoid reshuffling the order of the main linked list. The new list is sorted by the year born in ascending order. And then looped through displaying the UK passengers.