/\*

Name : Muhammad Nouman Zafar

ID : G00353137

Description : This program stores the details of the Passengers.And Gives user Different Options like

1. Add New Passenger

2. Display All Passengers

3. Display Passengers Details

4. Update Passengers Details

5. Delete Passenger

6. Generate Stats

7. Prints Details to File

8. Sort All Passengers from UK,Born before 1980

\*/

#include <stdio.h>

#include <string.h>

#include <conio.h>

//max length for the password

//Declairing valiables and arrays to store the stats

#define maxLength 20

int found = 0;

int passportNum = 0;

int travelClassCount[4];

int UK[4];

int EUROPE[4];

int ASIA[4];

int AUS[4];

int AMERICAS[4];

int oneDay[4];

int less3[4];

int less7[4];

int more7[4];

int totalPassengers = 0;

int bornYearStats[9];

int yearBornTotal = 0;

int passengerFound = 0;

//set head pointer to null

struct passengers\* headPointer = NULL;

struct passengers\* pass = NULL;

//user struct to get the user login and user verification

struct users

{

char username[20];

char pass[20];

struct users\* NEXT;

};

//passenger structure which holds the details of a passenger

struct passengers {

int passportNumber;

char firstName[20];

char secondName[20];

int yearBorn;

char emailAddress[50];

int travelFrom;

int travelClass;

int numOfTripsPerYear;

int avgStayDuration;

struct passengers\* NEXT;

};

//Declaring prototype of functions

void userLogin();

void addUsersFromFile(struct users\*\* top);

void addUsersInList(struct users\*\* top, char name[20], char pass[20]);

void searchUsers(struct users\* top, char name[20], char pass[20]);

void menu();

void passangerDetails(struct passengers\* newPassenger, int update);

void travelFrom();

void travelClass();

void numberOfTrips();

void stayDuration();

void displayAllPassengers(struct passengers\* top);

int lenght(struct passengers\* top);

int position(struct passengers\* top, int num);

void deletePassengerAtStart(struct passengers\*\* top);

void deletePassengerAtEnd(struct passengers\* top);

void deletePassenger(struct passengers\* top, int position);

void searchPassanger(struct passengers\* top, int passportNumber);

void displayPassengerDetails(struct passengers\* top);

void addPassengers(struct passengers\*\* top);

void printTravelFrom(struct passengers\* top);

void printTravelClass(struct passengers\* top);

void printTripsPerYear(struct passengers\* top);

void printAverageStay(struct passengers\* top);

void updatePassengerDetails(struct passengers\* top);

void countTravelFrom(struct passengers\* top, int check, int country[4], int travelFromm);

void stayDurationCheck(struct passengers\* top, int check, int stayy[4], int duration);

void countTravelClass(struct passengers\* top, int check);

void printStats(struct passengers\* top);

void print(int x);

void yearBornStats(struct passengers\* top, int check);

void printToFile(struct passengers\* top);

void printReadableData(struct passengers\* top);

void addPassengersFromFile(struct passengers\*\* top);

void saveStats();

void getStats();

void sortPassengersFromUK(struct passengers\* top, struct passengers\*\* pass);

void printData(FILE\* file, int x);

//main function

void main() {

//verify user

userLogin();

}

void userLogin() {

//Call the stats function to get the the stats from a FILE

getStats();

addPassengersFromFile(&headPointer);

char password[maxLength + 1];

char character;

int charPosition = 0;

int numeOfLines = 0;

char name[20];

int option = 0;

struct users\* headPointer = NULL;

addUsersFromFile(&headPointer);

printf("Username : ");

scanf("%s", name);

printf("Password : ");

//while loop ends when user hits Enter Button

while (1) {

//ASCII valuse will be retuned by the getch() function and stored in character variable

character = getch();

//check for the Enter Button

//13 represents Enter Button in ASCII table

if (character == 13) {

break;

}

//8 represents Backspace Button in ASCII table

else if (character == 8) {

if (charPosition > 0) {

charPosition--;

password[charPosition] = '\0';

printf("\b \b");

}

}

//32 is Space Button and 9 is Tab Button

//Don't add space or tab in the password

else if (character == 32 || character == 9) {

continue;

}

else {

if (charPosition < maxLength) {

password[charPosition] = character;

charPosition++;

printf("\*");

}

else {

printf("\nInput exceeds the Max Length of password, Only first 20 characters will be considered\n");

break;

}

}

}

password[charPosition] = '\0';

if (strlen(password) == 0) {

printf("\nNo Password Entered\n");

}

else {

searchUsers(headPointer, name, password);

printf("\n");

//if user is found in the file

if (found == 1) {

printf("Successfully Logged in\n");

menu();

}

else {//otherwise

printf("Login Failed\n");

printf("Choose one of the Following Options\n");

printf("\t1. Exit\n\t2. Try Again\n");

scanf("%d", &option);

while (option != 1) {

if (option == 2) {

userLogin();

}

else {

printf("Wrong Input\n");

}

printf("Choose one of the Following Options\n");

printf("\t1. Exit\n\t2. Try Again\n");

scanf("%d", &option);

}

}

}

}

//get user information from file

void addUsersFromFile(struct users\*\* top) {

FILE\* filePointer;

int num = 0;

char name[20];

char password[20];

//reading the data from file

filePointer = fopen("users.txt", "r");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

while (!feof(filePointer)) {

num = fscanf(filePointer, "%s %s", name, password);

if (num > 0) {

//read the information from file and add to Linked list

addUsersInList(top, name, password);

}

}//close the file

fclose(filePointer);

}

}

//add user function

void addUsersInList(struct users\*\* top, char name[20], char pass[20])

{

struct users\* newPassenger;

//create memory for new passenger

newPassenger = (struct users\*)malloc(sizeof(struct users) \* 1);

//copy given name amd password to the struct

strcpy(newPassenger->username, name);

strcpy(newPassenger->pass, pass);

newPassenger->NEXT = \*top;

\*top = newPassenger;

}

//serah the user in file wether user exst or not

void searchUsers(struct users\* top, char name[20], char pass[20]) {

struct users\* temp;

temp = top;

while (temp != NULL) {

//if user name and password match set found variable to 1 (true)

//and exit the loop

if ((strcmp(temp->username, name) == 0) && (strcmp(temp->pass, pass) == 0)) {

found = 1;

break;

}//otherwise keep looking for the user in file until the last node

else {

found = 0;

temp = temp->NEXT;

}

}

}

void menu() {

int option;

//give usr menu to run the application

printf("Choose one of the Following Options\n");

printf("\t0. Exit\n");

printf("\t1. Add Passenger\n\t2. Display all Passengers\n\t3. Display Passenger Details\n\t4. Update a Passenger statistic\n\t5. Delete Passenger\n\t6. Generate statistics\n");

printf("\t7. Print passenger Details to File\n\t8. Sort Uk Passengers in order of Year Born\n");

scanf("%d", &option);

while (option != 0) {

if (option == 1) {

//add passenger

int opt = 0;

printf("Enter the following Details of Passenger\n");

for (int j = 0; j < 1; j++) {

printf("\tPassport Number\t: ");

scanf("%d", &passportNum);

//search for the given passport number in the list

searchPassanger(headPointer, passportNum);

//if the given passport number already exists in the list

//give user option wheter change the passport number or

//delete the existing passenger and add new details with the passport number

if (passengerFound == 1) {

printf("Passenger Already Exists\n");

for (int i = 0; i < 1; i++) {

printf("\t1. Replace Existing Passenger\n\t2. Enter New Passport Number\n");

scanf("%d", &opt);

//if option is 1 then delete the existing passenger and add new details

if (opt == 1) {

int pos = position(headPointer, passportNum);

if (pos > 0) {

if (headPointer == NULL) {

printf("No Passenger Exists\n");

}

else if (pos < 2) {

deletePassengerAtStart(&headPointer);

}

else if (pos >= 2 && pos < (lenght(headPointer))) {

deletePassenger(headPointer, pos);

}

else {

deletePassengerAtEnd(headPointer);

}

}

addPassengers(&headPointer);

}//or otherwise give user a chance to enter a new passport number

else if (opt == 2) {

j--;

}

else {

i--;

}

}

}

else {

//if its a unique passport number then add a new passenger

//and add 1 to total Passengers in order to track the number of passengers in the list

addPassengers(&headPointer);

totalPassengers++;

passengerFound = 0;

}

}

}

else if (option == 2) {//if option is 2 then display the passport number and year they born

//to the screen

//display Passengers

printf("\n\n-----------------------------------------------------------+\n");

printf("\t\t All Passengers\n");

displayAllPassengers(headPointer);

}

else if (option == 3) {//display the details of a specific passenger

//Display Passenger Details

displayPassengerDetails(headPointer);

}

else if (option == 4) {//update details of existing passenger

//update passenger Details

updatePassengerDetails(headPointer);

}

else if (option == 5) {

//delete passenger

int num, opti;

for (int i = 0; i < 1; i++) {

if (headPointer == NULL) {

printf("\nNo Passenger in the Record\n\n");

break;

}

printf("Enter the Passport Number to Delete a Passenger : ");

scanf("%d", &num);

//get the position of given passport number

int pos = position(headPointer, num);

if (pos > 0) {//if position is greater than 0

//then delete the passenger

if (headPointer == NULL) {

printf("No Passenger Exists\n");

}

else if (pos < 2) {

deletePassengerAtStart(&headPointer);

}

else if (pos >= 2 && pos < (lenght(headPointer))) {

deletePassenger(headPointer, pos);

}

else {

deletePassengerAtEnd(headPointer);

}

totalPassengers--;

printf("\nPassenger Deleted\n");

}

else {//otherwise give user different options

//either enter a new passport number to detele the passenger

//or exit

printf("\nNo Passenger Exists of given Passport Number\n\n");

for (int j = 0; j < 1; j++) {

printf("Choose one of the following Options\n");

printf("\t1. Try Again\n\t2. Exit to Main Menu\n");

scanf("%d", &opti);

if (opti == 1) {

i--;

}

else if (opti == 2) {

break;

}

else {

printf("Wrong Input\n");

j--;

}

}

}

}

}

else if (option == 6) {

//print the Stats on the bases of the passengers exists in the list

printStats(headPointer);

}

else if (option == 7) {

//Print Passengers to File

if (headPointer == NULL) {

printf("\nNo Passenger in the Record\n\n");

}

else {

printReadableData(headPointer);

}

}

else if (option == 8) {

//Sort UK passengers in order of year Born

if (headPointer == NULL) {

printf("\nNo Passenger in the Record\n\n");

}

else {

sortPassengersFromUK(headPointer, &pass);

pass = NULL;

}

}

else {

printf("Wrong Input\n");

}

printf("Choose one of the Following Options\n");

printf("\t0. Exit\n");

printf("\t1. Add Passenger\n\t2. Display all Passengers\n\t3. Display Passenger Details\n\t4. Update a Passenger statistic\n\t5. Delete Passenger\n\t6. Generate statistics\n");

printf("\t7. Print passenger Details to File\n\t8. Sort Uk Passengers in order of Year Born\n");

scanf("%d", &option);

//whenever program is terminated

//the data is saved to file automatically

if (option == 0) {

//Print Passengers to File

if (headPointer == NULL) {

printf("\nNo Passenger in the Record\n\n");

}

else {

printToFile(headPointer);

saveStats();

printf("\nData Files are Updated\n\n");

}

}

}

}

//get the details of a passenger

void passangerDetails(struct passengers\* newPassenger, int update) {

//passport number

newPassenger->passportNumber = passportNum;

//first name

printf("\tFirst Name\t: ");

scanf("%s", newPassenger->firstName);

//last name

printf("\tSecond Name\t: ");

scanf("%s", newPassenger->secondName);

//year they were born

printf("\tYear Born\t: ");

scanf("%d", &newPassenger->yearBorn);

//email address

//email address should include

//@ and .com at the end

//otherwise keep aking the user to enter valid email

for (int i = 0; i < 1; i++) {

printf("\tEmail Address\t: ");

scanf("%s", newPassenger->emailAddress);

char atSign[2] = "@";

char dotCom[6] = ".com";

int valid = 0;

if (strstr(newPassenger->emailAddress, atSign) != NULL) {

if (strstr(newPassenger->emailAddress, dotCom) != NULL) {

valid = 1;

}

}

if (valid != 1) {

printf("Email Address is Not Valid\nMust Contain @ and .com\n");

i--;

}

}

for (int i = 0; i < 1; i++) {//travel from

printf("\tTravel From\n");

travelFrom();

scanf("%d", &newPassenger->travelFrom);

if (newPassenger->travelFrom < 1 || newPassenger->travelFrom > 5) {

printf("Wrong Input\n");

i--;

}

}

for (int i = 0; i < 1; i++) {//travel class

printf("\tTravel Class\n");

travelClass();

scanf("%d", &newPassenger->travelClass);

if (newPassenger->travelClass < 1 || newPassenger->travelClass>4) {

printf("Wrong Input\n");

i--;

}

}

for (int i = 0; i < 1; i++) {//number of trips a passenger makes per year

printf("\tNumber of trips per Year\n");

numberOfTrips();

scanf("%d", &newPassenger->numOfTripsPerYear);

if (newPassenger->numOfTripsPerYear < 1 || newPassenger->numOfTripsPerYear>3) {

printf("Wrong Input\n");

i--;

}

}

for (int i = 0; i < 1; i++) {//average time they stay

printf("\tStay Duration\n");

stayDuration();

scanf("%d", &newPassenger->avgStayDuration);

if (newPassenger->avgStayDuration < 1 || newPassenger->avgStayDuration>4) {

printf("Wrong Input\n");

i--;

}

}

if (update == 0) {//if these details are not used to update a passenger

//these functions are only used when new passenger is added

countTravelClass(newPassenger, 1);

yearBornStats(newPassenger, 1);

if (newPassenger->yearBorn < 1980) {

yearBornTotal++;

}

}

}

//user options to choose where they are traveling from

void travelFrom() {

printf("\t\t1. UK\n");

printf("\t\t2. Rest of Europe\n");

printf("\t\t3. Asia\n");

printf("\t\t4. Americas\n");

printf("\t\t5. Australasia\n");

}

//user options to choose what class they traveld

void travelClass() {

printf("\t\t1. Economy\n");

printf("\t\t2. Premium Economy\n");

printf("\t\t3. Business Class\n");

printf("\t\t4. First Class\n");

}

//how many trips passenger makes

void numberOfTrips() {

printf("\t\t1. Less than Three per year\n");

printf("\t\t2. Less than Five per year\n");

printf("\t\t3. More than Five per year\n");

}

//average stay

void stayDuration() {

printf("\t\t1. One Day\n");

printf("\t\t2. Less than Three days\n");

printf("\t\t3. Less than Seven days\n");

printf("\t\t4. More than Seven days\n");

}

//display all passenger to secreen

//display year they were born and their passport number

void displayAllPassengers(struct passengers\* top) {

struct passengers\* current;

int i = 0;

current = top;

printf("-----------------------------------------------------------+\n");

printf(" Passenger #\t Passport Number \t Year Born \n");

printf("-----------------------------------------------------------+\n");

//keep printing until the last passenger

while (current != NULL) {

printf("\t%d\t\t %d\t\t %d\n", i + 1, current->passportNumber, current->yearBorn);

current = current->NEXT;

i++;

}

printf("-----------------------------------------------------------+\n\n\n");

}

//update a existing passenger

void updatePassengerDetails(struct passengers\* top) {

int opti = 0;

int selection = 0;

char name[30];

int validateUpdateYear = 0;

for (int i = 0; i < 1; i++) {

struct passengers\* temp;

temp = top;

if (top == NULL) {

printf("\nNo Passenger in th Record\n\n");

break;

}

//give user 2 options

//either search by name

//or search by passport number

for (int k = 0; k < 1; k++) {

printf("Choose One of the Following Options\n");

printf("\t1. Search By Name\n\t2. Search By Passport Number\n");

scanf("%d", &selection);

if (selection == 1) {

//ask for th ename to search the passenger and update the details

printf("Enter the First Name to Update the Details : ");

scanf("%s", name);

}

else if (selection == 2) {

//ask the user to enter the passport number of the passenger they wants to update the details of

printf("Enter Passport Number to Update Passenger : ");

scanf("%d", &passportNum);

}

else {

printf("Wrong Input\n");

k--;

}

}

while (temp != NULL) {

//if name or the passport number matches

if (passportNum == temp->passportNumber || (strcmp(name, temp->firstName) == 0)) {//if passenger exists in the list

validateUpdateYear = temp->yearBorn;

//delete the previously added stats of the this passenger

yearBornStats(temp, 0);

countTravelClass(temp, 0);

passangerDetails(temp, 1);//ask for new details

if (validateUpdateYear != temp->yearBorn) {//if year born is not the same as last time the passmger was added

if (temp->yearBorn >= 1980) {

if (!(validateUpdateYear >= 1980)) {

yearBornTotal--;

}

}

else if (temp->yearBorn < 1980) {

if (!(validateUpdateYear < 1980)) {

yearBornTotal++;

}

}

}//add new stats

yearBornStats(temp, 1);

countTravelClass(temp, 1);

printf("Details Updated\n\n");

break;

}

else {

temp = temp->NEXT;

if (temp == NULL) {//if passenger does not exist in the list then give multiple options

//exit or enter new passport number

printf("\nNo Passenger Exists of given Passport Number OR First Name\n\n");

for (int j = 0; j < 1; j++) {

printf("Choose one of the following Options\n");

printf("\t1. Try Again\n\t2. Exit to Main Menu\n");

scanf("%d", &opti);

if (opti == 1) {

i--;

}

else if (opti == 2) {

break;

}

else {

printf("Wrong Input\n");

j--;

}

}

}

}

}

}

}

//search for the passnger in the list

void searchPassanger(struct passengers\* top, int passportNumber) {

struct passengers\* temp;

temp = top;

while (temp != NULL) {

if (passportNumber == temp->passportNumber) {//if passenger is iin the list

//det passenger found to 1 (true) and end the loop

passengerFound = 1;

break;

}

else {//otherwise keep searching until the last member of the list

passengerFound = 0;

temp = temp->NEXT;

}

}

}

//dislay all the details of the passenger

void displayPassengerDetails(struct passengers\* top) {

int passportNumber = 0;

int opti = 0;

int selection = 0;

char name[40];

for (int i = 0; i < 1; i++) {

struct passengers\* temp;

temp = top;

if (top == NULL) {

printf("\nNo Passenger in th Record\n\n");

break;

}

//give user 2 options

//either search by name

//or search by passport number

for (int k = 0; k < 1; k++) {

printf("Choose One of the Following Options\n");

printf("\t1. Search By Name\n\t2. Search By Passport Number\n");

scanf("%d", &selection);

if (selection == 1) {

//ask for the name

printf("Enter the First Name to see All Details : ");

scanf("%s", name);

}

else if (selection == 2) {

printf("Enter the Passport Number to see All Details : ");//ask for passport number to see the detais

scanf("%d", &passportNumber);

}

else {

printf("Wrong Input\n");

k--;

}

}

while (temp != NULL) {//search in the list until the last node of the list

//if name or the passport number matches

if (passportNumber == temp->passportNumber || (strcmp(name, temp->firstName) == 0)) {//if found display the details

printf("\n\nThese are the Details of the Passenger\n");

printf("\tPassport Number\t: %d\n", temp->passportNumber);

printf("\tFirst Name\t: %s\n", temp->firstName);

printf("\tSecond Name\t: %s\n", temp->secondName);

printf("\tYear Born\t: %d\n", temp->yearBorn);

printf("\tEmail address\t: %s\n", temp->emailAddress);

printTravelFrom(temp);

printTravelClass(temp);

printTripsPerYear(temp);

printAverageStay(temp);

printf("\n\n");

break;

}

else {//otherwise

temp = temp->NEXT;

if (temp == NULL) {//if passenger does not exist in the list then give multiple options

//exit or enter new passport number

printf("\nNo Passenger Exists of given Passport Number OR given Name\n\n");

for (int j = 0; j < 1; j++) {

printf("Choose one of the following Options\n");

printf("\t1. Try Again\n\t2. Exit to Main Menu\n");

scanf("%d", &opti);

if (opti == 1) {

i--;

}

else if (opti == 2) {

break;

}

else {

printf("Wrong Input\n");

j--;

}

}

}

}

}

}

}

//print passenger's average stay in Ireland on the bases of option they have choosen

void printAverageStay(struct passengers\* top) {

if (top->avgStayDuration == 1) {

printf("\tAverage Stay\t: %s\n", "One Day");

}

else if (top->avgStayDuration == 2) {

printf("\tAverage Stay\t: %s\n", "Less than 3 Days");

}

else if (top->avgStayDuration == 3) {

printf("\tAverage Stay\t: %s\n", "Less than 7 Days");

}

else if (top->avgStayDuration == 4) {

printf("\tAverage Stay\t: %s\n", "More than 7 Days");

}

else {

printf("\tAverage Stay\t: %s\n", "NOT VALID");

}

}

//number if trips in a year

void printTripsPerYear(struct passengers\* top) {

if (top->numOfTripsPerYear == 1) {

printf("\tTrips Per Year\t: %s\n", "Less Than Three Times");

}

else if (top->numOfTripsPerYear == 2) {

printf("\tTrips Per Year\t: %s\n", "Less Than Five Times");

}

else if (top->numOfTripsPerYear == 3) {

printf("\tTrips Per Year\t: %s\n", "More Than Five Times");

}

else {

printf("\tTrips Per Year\t: %s\n", "NOT VALID");

}

}

//travel class

void printTravelClass(struct passengers\* top) {

if (top->travelClass == 1) {

printf("\tTravel Class\t: %s\n", "Economy");

}

else if (top->travelClass == 2) {

printf("\tTravel Class\t: %s\n", "Premium Economy");

}

else if (top->travelClass == 3) {

printf("\tTravel Class\t: %s\n", "Business Class");

}

else if (top->travelClass == 4) {

printf("\tTravel Class\t: %s\n", "First Class");

}

else {

printf("\tTravel Class\t: %s\n", "NOT VALID");

}

}

//print where the passenger is travelling from

void printTravelFrom(struct passengers\* top) {

if (top->travelFrom == 1) {

printf("\tTraveled From\t: %s\n", "UK");

}

else if (top->travelFrom == 2) {

printf("\tTraveled From\t: %s\n", "Rest Of Europe");

}

else if (top->travelFrom == 3) {

printf("\tTraveled From\t: %s\n", "Asia");

}

else if (top->travelFrom == 4) {

printf("\tTraveled From\t: %s\n", "Americas");

}

else if (top->travelFrom == 5) {

printf("\tTraveled From\t: %s\n", "Australasia");

}

else {

printf("\tTraveled From\t: %s\n", "NOT VALID");

}

}

//calculate na dprint all the stats on the bases of passeneger in the list

void printStats(struct passengers\* top) {

int opt, option, op;

for (int i = 0; i < 1; i++) {

printf("Choose one of the following options\n");

printf("\t1. Stats using Travel Class\n\t2. Stats using Born Before 1980\n");

scanf("%d", &opt);

if (opt == 1) {

//travel stats

printf("\n\nStatistics of XYZ Airport\n");

printf("Total Passengers\t: %d\n", totalPassengers);//ask for the passenger

for (int a = 0; a < 1; a++) {//check the class

printf("Check Stats of...?\n");

printf("\t1. Economy Class\n");

printf("\t2. Premium Economy\n");

printf("\t3. Business Class\n");

printf("\t4. First Class\n");

scanf("%d", &option);

if (option == 1) {//print the stats for Economic Class

printf("Economy Class Statistics\n");

printf("\tEconomy Class Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[0] / (float)totalPassengers) \* 100, '%', travelClassCount[0]);

print(0);

}

else if (option == 2) {//print the stats for Premium Economic Class

printf("Premium Economy Statistics\n");

printf("\tPremium Economy Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[1] / (float)totalPassengers) \* 100, '%', travelClassCount[1]);

print(1);

}

else if (option == 3) {//print the stats for Business Class

printf("Business Class Statistics\n");

printf("\tBusiness Class Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[2] / (float)totalPassengers) \* 100, '%', travelClassCount[2]);

print(2);

}

else if (option == 4) {//print the stats for First Class

printf("First Class Statistics\n");

printf("\tFirst Class Passengers\t\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[3] / (float)totalPassengers) \* 100, '%', travelClassCount[3]);

print(3);

}

else {

printf("Wrong Input\n");

a--;

}

for (int c = 0; c < 1; c++) {

printf("\n\n");

printf("Check other Travel Class Stats ?\n");

printf("\t1. Yes\n\t2. No\n");

scanf("%d", &op);

if (op == 1) {

a--;

}

else if (op == 2) {

printf("\n\n");

break;

}

else {

printf("Wrong Input\n");

c--;

}

}

}

}

else if (opt == 2) {

//year born stats

//stats for the passengers who were born before 1980

printf("\n\nStats based on Passenger Born Before 1980\n");

printf("Statistics of XYZ Airport\n");

printf("Total Passengers\t: %d\n", totalPassengers);

printf("\tPassengers born before 1980\t: %.2f %c\t(%d Passenger)\n", ((float)yearBornTotal / (float)totalPassengers) \* 100, '%', yearBornTotal);

printf("\t\tUK\t\t\t\t: %.2f %c\n", ((float)bornYearStats[0] / (float)yearBornTotal) \* 100, '%');

printf("\t\tEurope\t\t\t\t: %.2f %c\n", ((float)bornYearStats[1] / (float)yearBornTotal) \* 100, '%');

printf("\t\tAsia\t\t\t\t: %.2f %c\n", ((float)bornYearStats[2] / (float)yearBornTotal) \* 100, '%');

printf("\t\tAmericas\t\t\t: %.2f %c\n", ((float)bornYearStats[3] / (float)yearBornTotal) \* 100, '%');

printf("\t\tAustralasia\t\t\t: %.2f %c\n", ((float)bornYearStats[4] / (float)yearBornTotal) \* 100, '%');

printf("\t\tOne Day Stay\t\t\t: %.2f %c\n", ((float)bornYearStats[5] / (float)yearBornTotal) \* 100, '%');

printf("\t\tLess than 3 Days\t\t: %.2f %c\n", ((float)bornYearStats[6] / (float)yearBornTotal) \* 100, '%');

printf("\t\tLess than 7 Days\t\t: %.2f %c\n", ((float)bornYearStats[7] / (float)yearBornTotal) \* 100, '%');

printf("\t\tMore than 7 Days\t\t: %.2f %c\n", ((float)bornYearStats[8] / (float)yearBornTotal) \* 100, '%');

printf("\n\n");

}

else {

printf("Wrong Input\n");

i--;

}

}

}

//printing readable data to the file

void printReadableData(struct passengers\* top) {

//file pointers to display different stats in different files

FILE\* filePointer;

FILE\* economy;

FILE\* Premium\_Economy;

FILE\* Business\_Class;

FILE\* First\_Class;

FILE\* Year\_Born;

int i = 0;

int option = 0, opt = 0, select = 0, op = 0;

struct passengers\* temp;

temp = top;

filePointer = fopen("Passenger\_Data.txt", "w");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

for (int j = 0; j < 1; j++) {//display all the details of all passengers

printf("Choose One of the Following Options\n");

printf("\t1. All Passengers Details\n\t2. Passenger Travel Statistics\n");

scanf("%d", &option);

if (option == 1) {

while (temp != NULL) {

fprintf(filePointer, "\nPassenger Number : %d\n", ++i);

fprintf(filePointer, "\tPassport # \t: %d\n\tFirst Name \t: %s\n\tLast Name\t: %s\n\tYear Born\t: %d\n\tEmail\t\t: %s\n\tTravel From\t: %d\n\tTravel Class\t: %d\n\tTrips Per Year\t: %d\n\tAverage Stay\t: %d\n",

temp->passportNumber,

temp->firstName,

temp->secondName,

temp->yearBorn,

temp->emailAddress,

temp->travelFrom,

temp->travelClass,

temp->numOfTripsPerYear,

temp->avgStayDuration);

temp = temp->NEXT;

}

printf("\nSaved To The File\n");

fclose(filePointer);

}

else if (option == 2) {//display the stats

//based on the travel class of the passenger year born

for (int k = 0; k < 1; k++) {

printf("Choose one of the following options\n");

printf("\t1. Stats using Travel Class\n\t2. Stats using Born Before 1980\n");

scanf("%d", &opt);

if (opt == 1) {

for (int a = 0; a < 1; a++) {

printf("Print Stats of...?\n");

printf("\t1. Economy Class\n");

printf("\t2. Premium Economy\n");

printf("\t3. Business Class\n");

printf("\t4. First Class\n");

scanf("%d", &select);

if (select == 1) {//Economy Class Stats

economy = fopen("Economy\_Stats.txt", "w");

fprintf(economy, "Statistics of XYZ Airport\n\n");

fprintf(economy, "Total Passengers\t: %d\n", totalPassengers);

fprintf(economy, "Economy Class Statistics\n");

fprintf(economy, "\tEconomy Class Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[0] / (float)totalPassengers) \* 100, '%', travelClassCount[0]);

printData(economy, 0);

fclose(economy);

printf("\nSaved To The File\n");

}

else if (select == 2) {//Premium economy Class Stats

Premium\_Economy = fopen("Premium\_Economy\_Stats.txt", "w");

fprintf(Premium\_Economy, "Statistics of XYZ Airport\n\n");

fprintf(Premium\_Economy, "Total Passengers\t: %d\n", totalPassengers);

fprintf(Premium\_Economy, "Premium Economy Statistics\n");

fprintf(Premium\_Economy, "\tPremium Economy Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[1] / (float)totalPassengers) \* 100, '%', travelClassCount[1]);

printData(Premium\_Economy, 1);

fclose(Premium\_Economy);

printf("\nSaved To The File\n");

}

else if (select == 3) {//Business Class Stats

Business\_Class = fopen("Business\_Class\_Stats.txt", "w");

fprintf(Business\_Class, "Statistics of XYZ Airport\n\n");

fprintf(Business\_Class, "Total Passengers\t: %d\n", totalPassengers);

fprintf(Business\_Class, "Business Class Statistics\n");

fprintf(Business\_Class, "\tBusiness Class Passengers\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[2] / (float)totalPassengers) \* 100, '%', travelClassCount[2]);

printData(Business\_Class, 2);

fclose(Business\_Class);

printf("\nSaved To The File\n");

}

else if (select == 4) {//First Class Stats

First\_Class = fopen("First\_Class\_Stats.txt", "w");

fprintf(First\_Class, "Statistics of XYZ Airport\n\n");

fprintf(First\_Class, "Total Passengers\t: %d\n", totalPassengers);

fprintf(First\_Class, "First Class Statistics\n");

fprintf(First\_Class, "\tFirst Class Passengers\t\t: %.2f %c\t(%d Passenger)\n", ((float)travelClassCount[3] / (float)totalPassengers) \* 100, '%', travelClassCount[3]);

printData(First\_Class, 3);

fclose(First\_Class);

printf("\nSaved To The File\n");

}

else {

printf("Wrong Input\n");

a--;

}

for (int c = 0; c < 1; c++) {

printf("\n\n");

printf("Check other Travel Class Stats ?\n");

printf("\t1. Yes\n\t2. No\n");

scanf("%d", &op);

if (op == 1) {

a--;

}

else if (op == 2) {

printf("\n\n");

break;

}

else {

printf("Wrong Input\n");

c--;

}

}

}

}

else if (opt == 2) {

Year\_Born = fopen("Year\_Born\_Stats.txt", "w");

//year born stats

//stats for the passengers who were born before 1980

fprintf(Year\_Born, "Stats based on Passenger Born Before 1980\n");

fprintf(Year\_Born, "Statistics of XYZ Airport\n");

fprintf(Year\_Born, "Total Passengers\t: %d\n", totalPassengers);

fprintf(Year\_Born, "\tPassengers born before 1980\t: %.2f %c\t(%d Passenger)\n", ((float)yearBornTotal / (float)totalPassengers) \* 100, '%', yearBornTotal);

fprintf(Year\_Born, "\t\tUK\t\t\t\t: %.2f %c\n", ((float)bornYearStats[0] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tEurope\t\t\t\t: %.2f %c\n", ((float)bornYearStats[1] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tAsia\t\t\t\t: %.2f %c\n", ((float)bornYearStats[2] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tAmericas\t\t\t: %.2f %c\n", ((float)bornYearStats[3] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tAustralasia\t\t\t: %.2f %c\n", ((float)bornYearStats[4] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tOne Day Stay\t\t\t: %.2f %c\n", ((float)bornYearStats[5] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tLess than 3 Days\t\t: %.2f %c\n", ((float)bornYearStats[6] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tLess than 7 Days\t\t: %.2f %c\n", ((float)bornYearStats[7] / (float)yearBornTotal) \* 100, '%');

fprintf(Year\_Born, "\t\tMore than 7 Days\t\t: %.2f %c\n", ((float)bornYearStats[8] / (float)yearBornTotal) \* 100, '%');

printf("\nSaved To The File\n");

fclose(Year\_Born);

}

else {

printf("Wrong Input\n");

k--;

}

}

}

}

}

}

void printData(FILE\* file, int x) {//function for calculating the stats for the passengers

fprintf(file, "\t\tUK\t\t\t: %.2f %c\n", ((float)UK[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tEurope\t\t\t: %.2f %c\n", ((float)EUROPE[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tAsia\t\t\t: %.2f %c\n", ((float)ASIA[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tAmericas\t\t: %.2f %c\n", ((float)AMERICAS[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tAustralasia\t\t: %.2f %c\n", ((float)AUS[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tOne Day Stay\t\t: %.2f %c\n", ((float)oneDay[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tLess Than 3 Days\t: %.2f %c\n", ((float)less3[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tLess Than 7 Days\t: %.2f %c\n", ((float)less7[x] / (float)travelClassCount[x]) \* 100, '%');

fprintf(file, "\t\tMore Than 7 Days\t: %.2f %c\n", ((float)more7[x] / (float)travelClassCount[x]) \* 100, '%');

}

void print(int x) {//function for calculating the stats for the passengers

printf("\t\tUK\t\t\t: %.2f %c\n", ((float)UK[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tEurope\t\t\t: %.2f %c\n", ((float)EUROPE[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tAsia\t\t\t: %.2f %c\n", ((float)ASIA[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tAmericas\t\t: %.2f %c\n", ((float)AMERICAS[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tAustralasia\t\t: %.2f %c\n", ((float)AUS[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tOne Day Stay\t\t: %.2f %c\n", ((float)oneDay[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tLess Than 3 Days\t: %.2f %c\n", ((float)less3[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tLess Than 7 Days\t: %.2f %c\n", ((float)less7[x] / (float)travelClassCount[x]) \* 100, '%');

printf("\t\tMore Than 7 Days\t: %.2f %c\n", ((float)more7[x] / (float)travelClassCount[x]) \* 100, '%');

}

void countTravelClass(struct passengers\* top, int check) {//calculate the stats based on the travel class

//check whether passenger is being added or updated or deleted

//1 in case of adding a new passenger

//0 if passenger is updated or deleted

countTravelFrom(top, check, UK, 1);

countTravelFrom(top, check, ASIA, 3);

countTravelFrom(top, check, AUS, 5);

countTravelFrom(top, check, AMERICAS, 4);

countTravelFrom(top, check, EUROPE, 2);

stayDurationCheck(top, check, oneDay, 1);

stayDurationCheck(top, check, less3, 2);

stayDurationCheck(top, check, less7, 3);

stayDurationCheck(top, check, more7, 4);

if (top->travelClass == 1) {

if (check == 1) {

travelClassCount[0]++;

}

else if (check == 0) {

travelClassCount[0]--;

}

}

else if (top->travelClass == 2) {

if (check == 1) {

travelClassCount[1]++;

}

else if (check == 0) {

travelClassCount[1]--;

}

}

else if (top->travelClass == 3) {

if (check == 1) {

travelClassCount[2]++;

}

else if (check == 0) {

travelClassCount[2]--;

}

}

else if (top->travelClass == 4) {

if (check == 1) {

travelClassCount[3]++;

}

else if (check == 0) {

travelClassCount[3]--;

}

}

}

//check the duration passenegr stay in Ireland

//and the travel class they travelled

void stayDurationCheck(struct passengers\* top, int check, int stayy[4], int duration) {

if (top->avgStayDuration == duration && top->travelClass == 1) {

if (check == 1) {

stayy[0]++;

}

else if (check == 0) {

stayy[0]--;

}

}

else if (top->avgStayDuration == duration && top->travelClass == 2) {

if (check == 1) {

stayy[1]++;

}

else if (check == 0) {

stayy[1]--;

}

}

else if (top->avgStayDuration == duration && top->travelClass == 3) {

if (check == 1) {

stayy[2]++;

}

else if (check == 0) {

stayy[2]--;

}

}

else if (top->avgStayDuration == duration && top->travelClass == 4) {

if (check == 1) {

stayy[3]++;

}

else if (check == 0) {

stayy[3]--;

}

}

}

//calculate the stats based upon the year a passenger is born

void yearBornStats(struct passengers\* top, int check) {

if (top->yearBorn < 1980 && top->travelFrom == 1) {

if (check == 1) {

bornYearStats[0]++;

}

else if (check == 0) {

bornYearStats[0]--;

}

}

else if (top->yearBorn < 1980 && top->travelFrom == 2) {

if (check == 1) {

bornYearStats[1]++;

}

else if (check == 0) {

bornYearStats[1]--;

}

}

else if (top->yearBorn < 1980 && top->travelFrom == 3) {

if (check == 1) {

bornYearStats[2]++;

}

else if (check == 0) {

bornYearStats[2]--;

}

}

else if (top->yearBorn < 1980 && top->travelFrom == 4) {

if (check == 1) {

bornYearStats[3]++;

}

else if (check == 0) {

bornYearStats[3]--;

}

}

else if (top->yearBorn < 1980 && top->travelFrom == 5) {

if (check == 1) {

bornYearStats[4]++;

}

else if (check == 0) {

bornYearStats[4]--;

}

}

if (top->yearBorn < 1980 && top->avgStayDuration == 1) {

if (check == 1) {

bornYearStats[5]++;

}

else if (check == 0) {

bornYearStats[5]--;

}

}

else if (top->yearBorn < 1980 && top->avgStayDuration == 2) {

if (check == 1) {

bornYearStats[6]++;

}

else if (check == 0) {

bornYearStats[6]--;

}

}

else if (top->yearBorn < 1980 && top->avgStayDuration == 3) {

if (check == 1) {

bornYearStats[7]++;

}

else if (check == 0) {

bornYearStats[7]--;

}

}

else if (top->yearBorn < 1980 && top->avgStayDuration == 4) {

if (check == 1) {

bornYearStats[8]++;

}

else if (check == 0) {

bornYearStats[8]--;

}

}

}

//check and calculate the stats on the bases of passenger has traveled and the class they were in e.g economy class

void countTravelFrom(struct passengers\* top, int check, int country[4], int travelFromm) {

if (top->travelFrom == travelFromm && top->travelClass == 1) {

if (check == 1) {

country[0]++;

}

else if (check == 0) {

country[0]--;

}

}

else if (top->travelFrom == travelFromm && top->travelClass == 2) {

if (check == 1) {

country[1]++;

}

else if (check == 0) {

country[1]--;

}

}

else if (top->travelFrom == travelFromm && top->travelClass == 3) {

if (check == 1) {

country[2]++;

}

else if (check == 0) {

country[2]--;

}

}

else if (top->travelFrom == travelFromm && top->travelClass == 4) {

if (check == 1) {

country[3]++;

}

else if (check == 0) {

country[3]--;

}

}

}

//get the length of the list

int lenght(struct passengers\* top) {

struct passengers\* temp;

int len = 0;

temp = top;

while (temp != NULL) {

len++;

temp = temp->NEXT;

}

return len;

}

//get the position of the the passenger in the list

//based on the passport number is given

int position(struct passengers\* top, int num) {

struct passengers\* temp;

temp = top;

int pos = 1;

while (temp != NULL) {

if (num == temp->passportNumber) {//if passport number is found end the loop

break;

}//otherwise keep going

pos++;

temp = temp->NEXT;

if (pos > lenght(top)) {//if there is no given passport number in the list then return -1 as result

pos = -1;

}

}

return pos;

}

//delete a passenger at given position

void deletePassenger(struct passengers\* top, int position) {

struct passengers\* current;

struct passengers\* prev\_current;

int i;

for (i = 0; i < position - 1; i++) {

prev\_current = current;

current = current->NEXT;

}

prev\_current->NEXT = current->NEXT;

if (current->yearBorn < 1980) {//if passenger was born before 1980 decrease 1 from the yearBornTotal

yearBornTotal--;

}

//delete the stats of the passenger

yearBornStats(current, 0);

countTravelClass(current, 0);

free(current);

}

//delete passenger at start of teh list

void deletePassengerAtStart(struct passengers\*\* top) {

struct passengers\* temp;

temp = \*top;

\*top = temp->NEXT;

if (temp->yearBorn < 1980) {

yearBornTotal--;

}

yearBornStats(temp, 0);

countTravelClass(temp, 0);

free(temp);

}

//delete passenger at the end of the list

void deletePassengerAtEnd(struct passengers\* top) {

struct passengers\* current;

struct passengers\* prev\_current;

current = top;

while (current->NEXT != NULL) {//look for the passenger until the end of the list

prev\_current = current;

current = current->NEXT;

}

prev\_current->NEXT = NULL;

if (current->yearBorn < 1980) {

yearBornTotal--;

}

yearBornStats(current, 0);

countTravelClass(current, 0);

free(current);//free the memory of the deleted passenger

}

//add a new passenger in the list

//in a sorted order

//based on the passport number

void addPassengers(struct passengers\*\* top) {

struct passengers\* newPassenger;

newPassenger = (struct passengers\*)malloc(sizeof(struct passengers) \* 1);

struct passengers\* temp;

temp = \*top;

//ask for user details

passangerDetails(newPassenger, 0);

//if list is empty

if (\*top == NULL) {

\*top = newPassenger;

newPassenger->NEXT = NULL;

}

else {//if list is not empty

//if new passenger passport number is less than the passport number exists in the list

if (newPassenger->passportNumber < temp->passportNumber) {

newPassenger->NEXT = temp;

\*top = newPassenger;

}

else {

while (temp->NEXT != NULL && temp->NEXT->passportNumber < newPassenger->passportNumber) {

temp = temp->NEXT;

}

newPassenger->NEXT = temp->NEXT;

temp->NEXT = newPassenger;

}

}

}

//sort the All UK passengers based on the year they were born

void sortPassengersFromUK(struct passengers\* top, struct passengers\*\* first) {

struct passengers\* newPassenger;

struct passengers\* temp;

struct passengers\* current;

temp = top;

while (temp != NULL) {//go over all the passegers

//and add the passenger in the list who wwre born before 1980

if (temp->yearBorn <= 1980 && temp->travelFrom == 1) {

newPassenger = (struct passengers\*)malloc(sizeof(struct passengers) \* 1);

current = \*first;

newPassenger->avgStayDuration = temp->avgStayDuration;

strcpy(newPassenger->emailAddress, temp->emailAddress);

strcpy(newPassenger->firstName, temp->firstName);

newPassenger->numOfTripsPerYear = temp->numOfTripsPerYear;

newPassenger->passportNumber = temp->passportNumber;

strcpy(newPassenger->secondName, temp->secondName);

newPassenger->travelClass = temp->travelClass;

newPassenger->travelFrom = temp->travelFrom;

newPassenger->yearBorn = temp->yearBorn;

if (\*first == NULL) {//add start of the list

\*first = newPassenger;

newPassenger->NEXT = NULL;

}

else {//sort the passengers

if (newPassenger->yearBorn < current->yearBorn) {

newPassenger->NEXT = current;

\*first = newPassenger;

}

else {//sort the passengers according to the year they wre born

while (current->NEXT != NULL&&current->NEXT->yearBorn < newPassenger->yearBorn) {

current = current->NEXT;

}

newPassenger->NEXT = current->NEXT;

current->NEXT = newPassenger;

}

}

temp = temp->NEXT;

}

else {

temp = temp->NEXT;

}

}

//if the list is empty give user a message

if (pass == NULL) {

printf("\n\nNo Passenger From Uk Born before 1980\n\n");

}

else {//otherwise

//display the sorted passengers

printf("\n\n-----------------------------------------------------------+\n");

printf("\t\t UK Passengers\n");

//call display Passengers fuction to display passengers who were born before 1980 and from UK

displayAllPassengers(pass);

}

}

//go over all the passengers and print the details of the passengers to a file

void printToFile(struct passengers\* top) {

FILE\* filePointer;

struct passengers\* temp;

temp = top;

filePointer = fopen("Passengers.txt", "w");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

while (temp != NULL) {

fprintf(filePointer, "%d %s %s %d %s %d %d %d %d\n",

temp->passportNumber,

temp->firstName,

temp->secondName,

temp->yearBorn,

temp->emailAddress,

temp->travelFrom,

temp->travelClass,

temp->numOfTripsPerYear,

temp->avgStayDuration);

temp = temp->NEXT;

}

fclose(filePointer);

}

}

//read back the data of passengers from file and to the list

void addPassengersFromFile(struct passengers\*\* top) {

int num = 0;

struct passengers\* passenger;

struct passengers\* temp;

FILE\* file;

file = fopen("Passengers.txt", "r");

if (file == NULL) {

printf("No Previous Passengers Records\n");

}

else {

while (!feof(file)) {

passenger = (struct passengers\*)malloc(sizeof(struct passengers) \* 1);

num = fscanf(file, "%d %s %s %d %s %d %d %d %d\n",

&passenger->passportNumber,

passenger->firstName,

passenger->secondName,

&passenger->yearBorn,

passenger->emailAddress,

&passenger->travelFrom,

&passenger->travelClass,

&passenger->numOfTripsPerYear,

&passenger->avgStayDuration);

if (num > 0) {

if (\*top == NULL) {

\*top = passenger;

passenger->NEXT = NULL;

}

else {

temp = \*top;

while (temp->NEXT != NULL) {

temp = temp->NEXT;

}

temp->NEXT = passenger;

passenger->NEXT = NULL;

}

}

}

fclose(file);//close the file

}

}

//save the stats of the passengers to the file

void saveStats() {

FILE\* filePointer;

filePointer = fopen("stats.txt", "w");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

for (int i = 0; i <= 9; i++) {

fprintf(filePointer, "%d\n", bornYearStats[i]);

}

fprintf(filePointer, "%d %d\n", totalPassengers, yearBornTotal);

for (int i = 0; i <= 4; i++) {

fprintf(filePointer, "%d %d %d %d %d %d %d %d %d %d\n",

travelClassCount[i],

UK[i],

EUROPE[i],

ASIA[i],

AUS[i],

AMERICAS[i],

oneDay[i],

less3[i],

less7[i],

more7[i]);

}

fclose(filePointer);

}

}

//read back all the stats from file

void getStats() {

FILE\* filePointer;

filePointer = fopen("stats.txt", "r");

if (filePointer != NULL) {

while (!feof(filePointer)) {

for (int i = 0; i <= 9; i++) {

fscanf(filePointer, "%d\n", &bornYearStats[i]);

}

fscanf(filePointer, "%d %d\n", &totalPassengers, &yearBornTotal);

for (int i = 0; i <= 4; i++) {

fscanf(filePointer, "%d %d %d %d %d %d %d %d %d %d\n",

&travelClassCount[i],

&UK[i],

&EUROPE[i],

&ASIA[i],

&AUS[i],

&AMERICAS[i],

&oneDay[i],

&less3[i],

&less7[i],

&more7[i]);

}

}

fclose(filePointer);

}

}