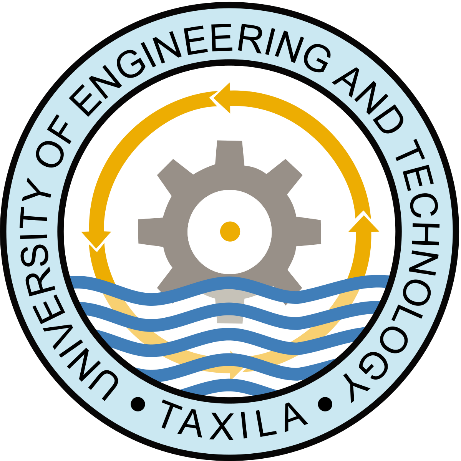
University of Engineering and Technology



Computer Programming Lab

Project Code

Submitted to:

Sir Amir Arsalan

Submitted by:

Raees Sattar (17-CP-01)

Asad Hameed (17-CP-23)

**Files Include in the Project**

1. Airline
2. Askclass.cpp
3. Asktransition.cpp
4. PTnamespace.cpp
5. CBnamespace.cpp
6. APnamespace.cpp
7. Initialization.cpp
8. SRnamespace.cpp
9. STnamespace.cpp

**File 1- Airline**

#include <iostream>

#include <iomanip>

#include <time.h>

#include <fstream>

#include <sstream>

using namespace std;

#include "Askclass.cpp"

#include "Asktransition.cpp"

#include "PTnamespace.cpp"

#include "CBnamespace.cpp"

#include "APnamespace.cpp"

#include "Initialization.cpp"

#include "SRnamespace.cpp"

#include "STnamespace.cpp"

int main()

{

/\*

This is variable is of a specific type and is responsible to display current

time on the top right corner of the program screen.

\*/

time\_t current\_time = time(0);

/\*

Assigning the names of 'Current Location, 'Airline',

'Destinations', 'Timmings for Flight', 'Class names ofr flight',

To provide the ease of editing if needed.

Furthermore, this assigning helps in generalizing

the code adn creating one module for diferent cases

\*/

string CurrentCity = "ISLAMABAD";

string CompanyName = "UET AIR";

string DestinationCity[4] = {"LAHORE" , "KARACHI" , "PESHAWAR" , "QUETTA"};

string FlightTimmings[4] = {"9:00 AM" , "11:00 AM" , "1:00 PM" , "3:00 AM"};

string Classes[3] = {"ECONOMY CLASS" , "FIRST CLASS" , "BUSINESS CLASS"};

int service, Destination, FlightTime, ClassType, ClassTransition;

int SeatCount[4][4][3] ;

/\*

Variable array CustomerInfo, stores the information about the Customer

Last entry '5' in the array shows that, Customer has to fill five blanks in order to book his/her seat.

Second last Entry '5' shows the numbers of seats for which the Customer has to provide information

\*/

string CustomerInfo[4][4][3][5][5];

string Customer[5] = {"FULL NAME " , "AGE " , " CNIC " , "PASSPORT NUMBER", "PASSWORD " };

/\*

Variable that stores input from the user to cancel his/her seat

\*/

int seat\_num;

/\*

Variable to count the number of times admin has logged in to his account,

to decide whether to take him to set the credentials or just login straight

\*/

int AdminLogin = 0;

string Admin[2], AdminCheck[2];

/\*

This variable aray is the core of our program. It stores information about the seat bookings.

Entry '1' shows the booked seat while '0' is for unbooked seat.

first Entry '4' shows the number of destinations.

Second Entry '4' Shows the number of flights that will fly on a certain time for a specific destionation.

third Entry '3' shows the number of Classes each flight has.

fourth Entry '5' Shows the number of seats each class contains.

\*/

int FlightInfo[4][4][3][5];

/\*Declaring a string to take input fro the fles conatining 'SeatCount'

and then converting that to integer values to store in array 'SeatCount'

Delaring second string for 'FlightInfo' for the same reason as above

\*/

string SC[4][4][3];

/\*Code to take input fro the files containing the 'SeactCount'

Advantage: Whenever the program runs, tit proceeds based on the previous data

\*/

ifstream Lahore1;

Lahore1.open("Countlahore.csv");

ifstream Karachi1;

Karachi1.open("Countkarachi.csv");

ifstream Peshawar1;

Peshawar1.open("Countpeshawar.csv");

ifstream Quetta1;

Quetta1.open("Countquetta.csv");

while(Lahore1.good())

{

for(int z = 0; z <= 3; z++)

{

getline(Lahore1, SC[0][z][0]);

getline(Lahore1, SC[0][z][0],',');

getline(Lahore1, SC[0][z][0]);

//

getline(Lahore1, SC[0][z][1]);

getline(Lahore1, SC[0][z][1],',');

getline(Lahore1, SC[0][z][1]);

//

getline(Lahore1, SC[0][z][2]);

getline(Lahore1, SC[0][z][2],',');

getline(Lahore1, SC[0][z][2]);

//Karachi

getline(Karachi1, SC[1][z][0]);

getline(Karachi1, SC[1][z][0],',');

getline(Karachi1, SC[1][z][0]);

//

getline(Karachi1, SC[1][z][1],',');

getline(Karachi1, SC[1][z][1]);

//

getline(Karachi1, SC[1][z][2],',');

getline(Karachi1, SC[1][z][2]);

//Peshawar

getline(Peshawar1, SC[2][z][0]);

getline(Peshawar1, SC[2][z][0],',');

getline(Peshawar1, SC[2][z][0]);

//

getline(Peshawar1, SC[2][z][1],',');

getline(Peshawar1, SC[2][z][1]);

//

getline(Peshawar1, SC[2][z][2],',');

getline(Peshawar1, SC[2][z][2]);

//Quetta

getline(Quetta1, SC[3][z][0]);

getline(Quetta1, SC[3][z][0],',');

getline(Quetta1, SC[3][z][0]);

//

getline(Quetta1, SC[3][z][1],',');

getline(Quetta1, SC[3][z][1]);

//

getline(Quetta1, SC[3][z][2],',');

getline(Quetta1, SC[3][z][2]);

}

}

Lahore1.close();

Karachi1.close();

Peshawar1.close();

Quetta1.close();

for(int abc = 0; abc <= 3; abc++)

{

for(int xyz = 0; xyz <=3; xyz++)

{

for(int jkl = 0; jkl <= 2; jkl++)

{

stringstream geek(SC[abc][xyz][jkl]);

geek >>SeatCount[abc][xyz][jkl];

}

}

}

IFI::InitialFightInfo(FlightInfo);

while(service != 5)

{

again\_service:

cout <<CurrentCity<<setw(50)<<"WELCOME TO "<<CompanyName<<setw(52)<<ctime(&current\_time)<<endl;

cout <<endl<<endl<<endl<<endl<<endl;

cout <<setw(72)<<"WE OFFER THE FOLLOWING SERVICES!"<<endl<<endl<<endl;

cout <<setw(80)<<"1. Login As Admin To Check For The Bookings"<<endl<<endl

<<setw(50)<<"2. Book A Seat"<<endl<<endl

<<setw(58)<<"3. Cancel Your Booking"<<endl<<endl

<<setw(56)<<"4. Print Your Ticket"<<endl<<endl

<<setw(43)<<"5. Exit"<<endl<<endl;

cout <<endl<<endl;

cout <<setw(87)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A SERVCIE!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>service;

cout <<endl<<endl;

switch(service)

{

case 1 :

AP::AdminPortal(AdminLogin, Admin, AdminCheck, DestinationCity,

FlightTimmings, Classes, Destination, FlightTime, FlightInfo);

break;

case 2 :

again\_destination:

cout <<setw(64)<<"DESTINATIONS "<<endl<<endl<<endl

<<setw(40)<<"1. "<<DestinationCity[0]<<endl

<<setw(40)<<"2. "<<DestinationCity[1]<<endl

<<setw(40)<<"3. "<<DestinationCity[2]<<endl

<<setw(40)<<"4. "<<DestinationCity[3]<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CITY!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>Destination;

cout <<endl<<endl<<endl;

//SWICTH FOR DESTINATION

switch(Destination)

{

case 1 :

case 2 :

case 3 :

case 4 :

ST::SelectingTime( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName, ClassTransition );

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CITY ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_destination;

}

break;

case 3 :

CB::CancelBooking( DestinationCity, FlightTimmings, Classes, Destination,

FlightTime, ClassType, FlightInfo, SeatCount);

break;

break;

case 4 :

PT::PrintTicket(CurrentCity, CompanyName, DestinationCity, FlightTimmings, Classes,

Destination, FlightTime, ClassType, FlightInfo, SeatCount, Customer, CustomerInfo);

break;

case 5:

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A SERVICE ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_service;

}

}

/\*

Storing the values of seat count of different cities to keep the program up to date when whenever it runs

\*/

ofstream Countlahore;

Countlahore.open("countlahore.csv");

for(int ad = 0; ad <= 3; ad++ )

{

Countlahore <<","<<","<<FlightTimmings[ad]<<","<<","<<endl;

for(int as = 0; as <= 2; as++)

{

Countlahore <<Classes[as]<<",";

Countlahore <<SeatCount[0][ad][as]<<endl;

}

}

//Karachi

ofstream Countkarachi;

Countkarachi.open("countkarachi.csv");

for(int a = 0; a <= 3; a++ )

{

Countkarachi <<","<<","<<FlightTimmings[a]<<","<<","<<endl;

for(int b = 0; b <= 2; b++)

{

Countkarachi <<Classes[b]<<",";

Countkarachi <<SeatCount[1][a][b]<<endl;

}

}

//Peshawar

ofstream Countpeshawar;

Countpeshawar.open("countpeshawar.csv");

for(int d = 0; d <= 3; d++ )

{

Countpeshawar <<","<<","<<FlightTimmings[d]<<","<<","<<endl;

for(int e = 0; e <= 2; e++)

{

Countpeshawar <<Classes[e]<<",";

Countpeshawar <<SeatCount[2][d][e]<<endl;

}

}

//Quetta

ofstream Countquetta;

Countquetta.open("countquetta.csv");

for(int g = 0; g <= 3; g++ )

{

Countquetta <<","<<","<<FlightTimmings[g]<<","<<","<<endl;

for(int h = 0; h <= 2; h++)

{

Countquetta <<Classes[h]<<",";

Countquetta <<SeatCount[3][g][h]<<endl;

}

}

return 0;

}

**File 2- AskClass**

namespace AskC

{

//ASKING CLASS TYPE

void AskClass(int ClassType, string Classes[4])

{

cout <<setw(68)<<"CLASSES WE OFFER"<<endl<<endl<<endl

<<setw(41)<<"1. "<<Classes[0]<<endl

<<setw(41)<<"2. "<<Classes[1]<<endl

<<setw(41)<<"3. "<<Classes[2]<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CLASS";

cout <<endl<<endl<<endl;

}

}

**File 3 – AskTransition**

namespace AskT

{

//ASKING TRANSITION

void AskTransition(int ClassTransition)

{

cout <<setw(66)<<"1. CHOOSE A DIFFERENT CLASS"<<endl

<<setw(66)<<"2. CHOOSE A DIFFERENT TIME "<<endl

<<setw(47)<<"3. QUIT "<<endl;

cout <<endl<<endl<<endl;

}

}

**File 4- PTnamespace**

namespace PT

{

void PrintTicket(string CurrentCity, string CompanyName, string DestinationCity[4], string FlightTimmings[4], string Classes[3],

int Destination, int FlightTime, int ClassType, int FlightInfo[4][4][3][5],

int seats\_count[4][4][3], string Customer[5], string CustomerInfo[4][4][3][5][5] )

{

/\*

Variable that stores input from the user to print his/her ticket

\*/

int seat\_num;

again\_ticket\_destination:

cout <<setw(93)<<"PLEASE PROVIDE THE THE FOLLOWING INFORMATION ABOUT YOUR SEAT BOOKING";

cout <<endl<<endl<<endl;

cout <<setw(64)<<"DESTINATIONS "<<endl<<endl<<endl

<<setw(40)<<"1. "<<DestinationCity[0]<<endl

<<setw(40)<<"2. "<<DestinationCity[1]<<endl

<<setw(40)<<"3. "<<DestinationCity[2]<<endl

<<setw(40)<<"4. "<<DestinationCity[3]<<endl

<<setw(40)<<"5. "<<"BACK"<<endl<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CITY!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>Destination;

cout <<endl<<endl<<endl;

switch(Destination)

{

case 1:

case 2:

case 3:

case 4:

again\_ticket\_timmings:

cout <<setw(64)<<"TIMMINGS"<<endl<<endl<<endl

<<setw(40)<<"1. "<<FlightTimmings[0]<<endl

<<setw(40)<<"2. "<<FlightTimmings[1]<<endl

<<setw(40)<<"3. "<<FlightTimmings[2]<<endl

<<setw(40)<<"4. "<<FlightTimmings[3]<<endl

<<setw(40)<<"5. "<<"BACK"<<endl<<endl<<endl;

cout <<setw(98)<<"PRESS THE CORRESPONDING NUMBER TO SELECT YOUR FLIGHT's TIME!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>FlightTime;

cout <<endl<<endl<<endl;

switch(FlightTime)

{

case 1 :

case 2 :

case 3 :

case 4 :

again\_ticket\_class:

cout <<setw(68)<<"CLASSES"<<endl<<endl<<endl

<<setw(41)<<"1. "<<Classes[0]<<endl

<<setw(41)<<"2. "<<Classes[1]<<endl

<<setw(41)<<"3. "<<Classes[2]<<endl

<<setw(41)<<"4. "<<"BACK"<<endl<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CLASS";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl;

switch(ClassType)

{

case 1 :

case 2 :

case 3 :

cout <<setw(64)<<"SEAT NUMBER"<<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>seat\_num;

cout <<endl<<endl<<endl;

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num-1] == 1)

{

cout <<setw(67)<<"YOUR TICKET";

cout <<endl<<endl;

cout <<setw(24) <<Customer[0]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num][0];

cout <<setw(24) <<"FROM "<<" "<<setw(25)<<CurrentCity;

cout <<endl<<endl;

cout <<setw(24) <<Customer[1]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num][1];

cout <<setw(24) <<"DESTINATION"<<" "<<setw(25)<<DestinationCity[Destination-1];

cout <<endl<<endl;

cout <<setw(24) <<Customer[2]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num][2];

cout <<setw(24) <<"FLIGHT TIME"<<" "<<setw(25)<<FlightTimmings[FlightTime-1];

cout <<endl<<endl;

cout <<setw(24) <<Customer[3]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num][3];

cout <<setw(24) <<"CLASS TYPE "<<" "<<setw(25)<<Classes[ClassType-1];

cout <<endl<<endl<<endl;

cout <<setw(70)<<"THANK YOU FOR CHOOSING "<<CompanyName;

cout <<endl<<endl<<endl<<endl<<endl;

break;

}

else

{

cout <<setw(70)<<"THIS SEAT IS NOT BOOKED."<<endl<<endl

<<setw(88)<<"PLEASE PROVIDE CORRECT INFORMATION TO PRINT YOUR TICKET"<<endl<<endl<<endl;

}

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ticket\_class;

}

break;

case 5 :

goto again\_ticket\_destination;

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A TIME ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ticket\_timmings;

}

break;

case 5 :

break;

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A DESTINATION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ticket\_destination;

}

}

}

**File 5- CBnamespace**

namespace CB

{

void CancelBooking(string DestinationCity[4], string FlightTimmings[4], string Classes[3],

int Destination, int FlightTime, int ClassType, int FlightInfo[4][4][3][5], int seats\_count[4][4][3])

{

/\*

Variable that stores input from the user to cancel his/her seat

\*/

int seat\_num;

again\_cancel\_destination:

cout <<setw(64)<<"DESTINATIONS "<<endl<<endl<<endl

<<setw(40)<<"1. "<<DestinationCity[0]<<endl

<<setw(40)<<"2. "<<DestinationCity[1]<<endl

<<setw(40)<<"3. "<<DestinationCity[2]<<endl

<<setw(40)<<"4. "<<DestinationCity[3]<<endl

<<setw(40)<<"5. "<<"BACK"<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CITY!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>Destination;

cout <<endl<<endl<<endl;

switch(Destination)

{

case 1 :

case 2 :

case 3 :

case 4 :

again\_cancel\_timmings:

cout <<setw(70)<<"TIMMINGS FOR FLIGHTS"<<endl<<endl<<endl

<<setw(40)<<"1. "<<FlightTimmings[0]<<endl

<<setw(40)<<"2. "<<FlightTimmings[1]<<endl

<<setw(40)<<"3. "<<FlightTimmings[2]<<endl

<<setw(40)<<"4. "<<FlightTimmings[3]<<endl

<<setw(40)<<"5. "<<"BACK"<<endl<<endl;

cout <<setw(98)<<"PRESS THE CORRESPONDING NUMBER TO SELECT YOUR FLIGHT's TIME!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>FlightTime;

cout <<endl<<endl;

switch(FlightTime)

{

case 1 :

case 2 :

case 3 :

case 4 :

again\_cancel\_class:

cout <<setw(68)<<"CLASSES WE OFFER"<<endl<<endl<<endl

<<setw(41)<<"1. "<<Classes[0]<<endl

<<setw(41)<<"2. "<<Classes[1]<<endl

<<setw(41)<<"3. "<<Classes[2]<<endl

<<setw(41)<<"4. "<<"BACK"<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CLASS";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl;

switch(ClassType)

{

case 1 :

case 2 :

case 3 :

cout <<setw(64)<<"SEAT NUMBER"<<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>seat\_num;

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num-1] == 1)

{

FlightInfo[Destination-1][FlightTime-1][ClassType-1][seat\_num-1] =0;

seats\_count[Destination-1][FlightTime-1][ClassType-1] -= 1;

cout <<FlightInfo[Destination-1][FlightTime-1][ClassType-1][seats\_count[Destination-1][FlightTime-1][ClassType-1]];

cout <<seats\_count[Destination-1][FlightTime-1][ClassType-1];

cout <<setw(77)<<"YOUR SEAT BOOKING HAS BEEN CANCELED"<<endl<<endl;

}

else

{

cout <<setw(85)<<"THIS SEAT IS NOT BOOKED."<<endl<<endl

<<setw(95)<<"PLEASE PROVIDE CORRECT INFORMATION TO CANCEL THE BOOKING"<<endl<<endl<<endl;

}

break;

case 4 :

goto again\_cancel\_timmings;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_cancel\_class;

}

break;

case 5 :

goto again\_cancel\_destination;

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A TIME ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_cancel\_timmings;

}

break;

case 5 :

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CITY ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_cancel\_destination;

}

}

}

**File 6 - APnamespace**

namespace AP

{

void AdminPortal(int &AdminLogin, string Admin[2], string AdminCheck[2], string DestinationCity[4], string FlightTimmings[4], string Classes[3],

int Destination, int FlightTime, int FlightInfo[4][4][3][5])

{

ifstream Fileadmin;

Fileadmin.open("admin.txt");

Fileadmin >>Admin[0];

Fileadmin >>Admin[1];

Fileadmin >>AdminLogin;

Fileadmin.close();

if(AdminLogin == 0)

{

cout <<setw(90)<<"CHOOSE A USER NAME ANE PASSWORD TO MAKE YOUR ACCOUNT SECURE"<<endl<<endl<<endl

<<setw(62)<<"USER NAME"<<endl<<endl<<endl;

cout <<setw(57)<<" ";

cin >>Admin[0];

cout <<endl<<endl;

cout <<setw(62)<<"PASSWORD"<<endl<<endl<<endl;

cout <<setw(57)<<" ";

cin >> Admin[1];

AdminLogin++;

ofstream Fileadmin1;

Fileadmin1.open("admin.txt");

Fileadmin1 <<Admin[0]<<endl;

Fileadmin1 <<Admin[1]<<endl;

Fileadmin1 <<AdminLogin;

}

else if(AdminLogin > 0 )

{

again\_admin:

cout <<setw(62)<<"USER NAME"<<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>AdminCheck[0];

cout <<endl<<endl;

cout <<setw(62)<<"PASSWORD"<<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>AdminCheck[1];

cout <<endl<<endl<<endl;

if(AdminCheck[0] == Admin[0] && AdminCheck[1] == Admin[1])

{

again\_admin\_destination:

cout <<setw(64)<<"DESTINATIONS "<<endl<<endl<<endl

<<setw(40)<<"1. "<<DestinationCity[0]<<endl

<<setw(40)<<"2. "<<DestinationCity[1]<<endl

<<setw(40)<<"3. "<<DestinationCity[2]<<endl

<<setw(40)<<"4. "<<DestinationCity[3]<<endl

<<setw(46)<<"5. LOGOUT"<<endl<<endl;

cout <<setw(85)<<"PRESS THE CORRESPONDING NUMBER TO SELECT A CITY!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>Destination;

cout <<endl<<endl<<endl;

switch(Destination)

{

case 1 :

case 2 :

case 3 :

case 4 :

again\_admin\_timmings:

cout <<setw(64)<<"TIMMINGS "<<endl<<endl<<endl

<<setw(40)<<"1. "<<FlightTimmings[0]<<endl

<<setw(40)<<"2. "<<FlightTimmings[1]<<endl

<<setw(40)<<"3. "<<FlightTimmings[2]<<endl

<<setw(40)<<"4. "<<FlightTimmings[3]<<endl

<<setw(40)<<"5. "<<"BACK"<<endl

<<setw(40)<<"6. "<<"LOGOUT"<<endl<<endl;

cout <<setw(98)<<"PRESS THE CORRESPONDING NUMBER TO SELECT YOUR FLIGHT's TIME!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>FlightTime;

cout <<endl<<endl<<endl;

switch(FlightTime)

{

case 1 :

case 2 :

case 3 :

case 4 :

cout <<setw(60)<<DestinationCity[Destination-1]<<endl<<endl;

cout <<setw(61)<<FlightTimmings[FlightTime-1]<<endl<<endl;

for(int i = 0; i <= 2; i++)

{

cout <<setw(55)<<Classes[i]<<" ";

for(int n = 0; n <= 4; n++)

{

cout <<FlightInfo[Destination-1][FlightTime-1][i][n]<<" ";

}

cout <<endl<<endl;

}cout <<endl<<endl;

break;

case 5 :

goto again\_admin\_destination;

case 6 :

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A TIME ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_admin\_timmings;

}

break;

case 5 :

break;

default:

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A DESTINATION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_admin\_destination;

}

}

else

{

cerr <<setw(75)<<"USERNAME OR PASSWORD IS INCORRECT!";

cout <<endl<<endl<<endl;

goto again\_admin;

}

}

}

}

**File 7 – Initialization**

namespace IFI

{

void InitialFightInfo(int FlightInfo[4][4][3][5])

{

for(int a = 0; a <= 3; a++)

{

for(int b = 0; b <= 3; b++)

{

for(int c = 0; c <= 2; c++)

{

for(int d = 0; d <= 4; d++)

{

FlightInfo[a][b][c][d] = 0;

}

}

}

}

}

}

namespace ISC

{

void InitialSeatCount(int SeatCount[4][4][3])

{

for(int a = 0; a <= 3; a++)

{

for(int b = 0; b <= 3; b++)

{

for(int c = 0; c <= 2; c++)

{

SeatCount[a][b][c] = 0;

}

}

}

}

}

**File 8 – Srnamepsace**

namespace SR

{

void SeatReservation(int Destination, int FlightTime, int ClassType, int FlightInfo[4][4][3][5], int SeatCount[4][4][3], string DestinationCity[4],

string FlightTimmings[4], string Classes[3], string CustomerInfo[4][4][3][5][5], string CurrentCity, string Customer[5], string CompanyName )

{

FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] = 1;

SeatCount[Destination-1][FlightTime-1][ClassType-1]++;

cout <<setw(52)<<"SEAT NUMBER "<<SeatCount[Destination-1][FlightTime-1][ClassType-1]<<" HAS BEEN RESERVED FOR YOU"<<endl<<endl;

//ASKING CUSTOMER FOR DATA

cout <<setw(87)<<"PLEASE PROVIDE THIS INFORMATION TO CONFIRM YOUR TICKET"<<endl<<endl<<endl;

cout <<setw(48)<<Customer[0]

<<setw(25)<<" ";

getline(cin, CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][0]);

getline(cin, CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][0]);

cout <<endl<<endl<<endl;

for(int c\_loop = 1; c\_loop <= 4; c\_loop++)

{

cout <<setw(48)<<Customer[c\_loop]

<<setw(25)<<" ";

getline(cin, CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][c\_loop]);

cout <<endl<<endl<<endl;

}

//PRINT TICKET

cout <<setw(67)<<"YOUR TICKET";

cout <<endl<<endl;

cout <<setw(24) <<Customer[0]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][0];

cout <<setw(24) <<"FROM "<<" "<<setw(25)<<CurrentCity;

cout <<endl<<endl;

cout <<setw(24) <<Customer[1]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][1];

cout <<setw(24) <<"DESTINATION"<<" "<<setw(25)<<DestinationCity[Destination-1];

cout <<endl<<endl;

cout <<setw(24) <<Customer[2]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][2];

cout <<setw(24) <<"FLIGHT TIME"<<" "<<setw(25)<<FlightTimmings[FlightTime-1];

cout <<endl<<endl;

cout <<setw(24) <<Customer[3]<<" "<<setw(25)<<CustomerInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]][3];

cout <<setw(24) <<"CLASS TYPE "<<" "<<setw(25)<<Classes[ClassType-1];

cout <<endl<<endl<<endl;

cout <<setw(70)<<"THANK YOU FOR CHOOSING "<<CompanyName;

cout <<endl<<endl<<endl<<endl<<endl;

}

}

**File 3 – Stnamespace**

namespace ST

{

void SelectingTime(int Destination, int FlightTime, int ClassType, int FlightInfo[4][4][3][5], int SeatCount[4][4][3], string DestinationCity[4],

string FlightTimmings[4], string Classes[3], string CustomerInfo[4][4][3][5][5], string CurrentCity, string Customer[5], string CompanyName, int ClassTransition )

{

again\_time:

again\_f\_timmings0:

cout <<setw(70)<<"TIMMINGS FOR FLIGHTS"<<endl<<endl<<endl

<<setw(40)<<"1. "<<FlightTimmings[0]<<endl

<<setw(40)<<"2. "<<FlightTimmings[1]<<endl

<<setw(40)<<"3. "<<FlightTimmings[2]<<endl

<<setw(40)<<"4. "<<FlightTimmings[3]<<endl<<endl;

cout <<setw(98)<<"PRESS THE CORRESPONDING NUMBER TO SELECT YOUR FLIGHT's TIME!";

cout <<endl<<endl<<endl;

cout <<setw(57)<<" ";cin >>FlightTime;

cout <<endl<<endl;

//SWITCH FOR FLIGHT TIME

switch(FlightTime)

{

case 1 :

again\_class00:

again\_f\_class000:

AskC::AskClass( ClassType, Classes );

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TYPE

switch(ClassType)

{

case 1 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

//ASKING CUSTOMER FOR DATA

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[0]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class000;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_fc\_transitions00;

} break;

}

else continue;

}

break;

case 2 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[1]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_fc\_transitions00:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class000;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_fc\_transitions00;

} break;

}

else continue;

}

break;

case 3 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

//ASKING CUSTOMER FOR DATA

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[2]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_bc\_transition00:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class000;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_bc\_transition00;

} break;

}

else continue;

}

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_class00;

}

break;

case 2 :

again\_class01:

again\_f\_class001:

AskC::AskClass( ClassType, Classes );

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TYPE

switch(ClassType)

{

case 1 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[0]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_ec\_transition01:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class001;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ec\_transition01;

} break;

}

else continue;

}

break;

case 2 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[1]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_fc\_transitions01:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class001;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_fc\_transitions01;

} break;

}

else continue;

}

break;

case 3 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[2]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_bc\_transition01:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class001;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_bc\_transition01;

} break;

}

else continue;

}

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_class01;

}

break;

case 3 :

again\_class02:

again\_f\_class002:

AskC::AskClass( ClassType, Classes );

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TYPE

switch(ClassType)

{

case 1 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[0]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_ec\_transition02:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class002;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ec\_transition02;

} break;

}

else continue;

}

break;

case 2 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[1]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_fc\_transitions02:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class002;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_fc\_transitions02;

} break;

}

else continue;

}

break;

case 3 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[2]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_bc\_transition02:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class002;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_bc\_transition02;

} break;

}

else continue;

}

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_class02;

}

break;

case 4 :

again\_class03:

again\_f\_class003:

AskC::AskClass( ClassType, Classes );

cout <<setw(57)<<" ";cin >>ClassType;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TYPE

switch(ClassType)

{

case 1 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1]>= 5)

{

cout <<setw(50)<<Classes[0]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_ec\_transition03:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class003;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_ec\_transition03;

} break;

}

else continue;

}

break;

case 2 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[1]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_fc\_transitions03:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class003;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_fc\_transitions03;

} break;

}

else continue;

}

break;

case 3 :

while(SeatCount[Destination-1][FlightTime-1][ClassType-1] <= 240)

{

if(FlightInfo[Destination-1][FlightTime-1][ClassType-1][SeatCount[Destination-1][FlightTime-1][ClassType-1]] == 0 && SeatCount[Destination-1][FlightTime-1][ClassType-1] < 5)

{

SR::SeatReservation( Destination, FlightTime, ClassType, FlightInfo, SeatCount, DestinationCity,

FlightTimmings, Classes, CustomerInfo, CurrentCity, Customer, CompanyName );

break;

}

else if(SeatCount[Destination-1][FlightTime-1][ClassType-1] >= 5)

{

cout <<setw(50)<<Classes[2]<<" FOR THIS FLIGHT HAS BEEN BOOKED COMPLETELY!"<<endl<<endl;

again\_bc\_transition03:

AskT::AskTransition( ClassTransition );

cout <<setw(57)<<" ";cin >>ClassTransition;

cout <<endl<<endl<<endl;

//SWITCH FOR CLASS TRANSITION

switch(ClassTransition)

{

case 1 :

goto again\_f\_class003;

break;

case 2 :

goto again\_f\_timmings0;

break;

case 3 :

break;

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE AN OPTION ONLY BY PRESSINGTHE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_bc\_transition03;

} break;

}

else continue;

}

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A CLASS ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_class03;

}

break;

default :

cerr <<setw(61)<<"ERROR!"<<endl

<<setw(98)<<"YOU CAN CHOOSE A TIME ONLY BY PRESSING THE NUMBER CORRESPONDING TO IT.";

cout <<endl<<endl<<endl;

goto again\_time;

}

}

}