|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Assessment Brief/Cover Sheet** | | | | | |
| **Class Group:** | | 062CS | | | |
| **Assessor:** | | Maura O’Halloran | | | |
| **Component Title and Code:** | | Object Oriented Programming, 6N2108 | | | |
| **Assessment Technique:** | | Skills Demo | | **Weighting:** | 30% |
|  | |  | | | |
| **Title:** | | Skills Demo #1 | | | |
| **Issue Date:** | | 11th January 2021 | | **Submission Date:** | 1st February 2021 |
| **Learning Outcomes Assessed:** | | LO2, LO3, L04 and LO5 | | | |
| **Guidelines:** Fully address each point in the requirements section of this brief. | | | | | |
| **Assessment Criteria** | | | **Available Marks** | | |
| Program design | | | 8 | | |
| Program implementation | | | 12 | | |
| Quality of application | | | 6 | | |
| Testing of application | | | 4 | | |
| **Learner Name:** | *Milan Labus* | | | | |
| I confirm that:   1. I have been provided with information about Cork ETB’s assessment and appeals procedures and my responsibilities with regard to assessment. 2. The assessment work produced by me is all my own original work. | | | | | |
| **Note to Learners:**   * Plagiarism is the presentation of someone else’s ideas, arguments, concepts or work as your own by failing to reference or acknowledge it properly. All such work must be acknowledged.   Any learner, who presents another’s work as their own, will be investigated in line with Cork ETB Assessment Malpractice procedures and may be awarded a zero grade.   * Learners should keep copies of all assessment submitted, where applicable. | | | | | |

**LDF TRAVEL – LONG DISTANCE FLIGHTS**

LDF Travel books long distance flights for customers. A long-distance journey, from source to destination, requires two flights. A single flight consists of:

1. A departure airport
2. A departure date
3. A departure time (hours and minutes)
4. An arrival airport
5. An arrival date
6. An arrival time (hours and minutes)
7. A journey length (hours and minutes)
8. A cost
9. A class (economy, premium economy, business or first class).

When a customer enters LDF Travel, they tell the travel agent their required departure and arrival date, the airport from which they want to depart and the class that they want to travel. The travel agent checks the available journey options and presents them to the customer. The customer decides which flight option suits them best and the travel agent enters the flight details and their name.

For example, the following two flights show a possible journey from Cork to Dubai. In this case, the customer leaves Cork at 10:45am on the 1st February 2020 and arrives in Dubai on the 2nd February at 20:15pm. The flight from Cork to Dubai takes a total of 33 hours and 30 minutes and costs €1229.94.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Leaving |  | Date | H | M | Arriving | Date | H | M | H | M | Cost |
| ORK |  | 01.02.2020 | 10 | 45 | GLA | 01.02.2020 | 12 | 15 | 1 | 30 | €129.99 |
|  |  |  |  |  |  |  |  |  |  |  |  |
| Leaving |  | Date | H | M | Arriving | Date | H | M | H | M | Cost |
| GLA |  | 02.02.2020 | 8 | 10 | DXB | 02.02.2020 | 20 | 15 | 12 | 5 | €1099.95 |

The two boarding passes comprising their journey are then printed in the following format:

BOARDING PASS: <Name> CLASS: <Class> COST: <Flight cost>

FROM DATE TIME <Departure Airport> <Departure Date> <Departure Time>

TO <Arrivals Airport> <Arrivals Date> <Arrivals Time>

# EXPECTED FLIGHT TIME: <Flight Time Hours>H <Flight Time Minutes>M

For example, the first boarding pass for the journey above will be printed as:

|  |  |  |  |
| --- | --- | --- | --- |
| BOARDING PASS: Mary Smith | CLASS: ECONOMY |  | COST: €129.99 |
| FROM | DATE |  | TIME |
| ORK | 01.02.2020 |  | 10:45 |
| TO  GLA    EXPECTED FLIGHT TIME: 1H 30M | 01.02.2020 |  | 12:15 |

The second boarding pass will be printed as:

|  |  |  |  |
| --- | --- | --- | --- |
| BOARDING PASS: Mary Smith | CLASS: ECONOMY |  | COST: €1099.95 |
| FROM | DATE |  | TIME |
| GLA | 02.02.2020 |  | 08:10 |
| TO  DXB    EXPECTED FLIGHT TIME: 12H 5M | 01.02.2020 |  | 20:15 |

When the boarding passes have been printed, the travel agent enters the customer’s name and address and generates the following summary letter.

LDF TRAVEL

Name: <Customer Name>

Address: <Address Line 1>

<Address Line 2>

<Address Line 3>

REF: <Departure Airport>/<Destination Airport>/<Arrival Date>

Dear <FirstName>,

Please find enclosed your boarding passes for your flight from <destination airport> to <arrival airport>.

Your journey is as follows:

SUMMARY OF JOURNEY

NAME: Mary Smith CLASS: ECONOMY COST: <Total flight cost>

FROM DATE TIME <First Departure Airport> <First Departure Date> <First Departure Time>

TO <Final Arrival Airport> <Final Arrival Date> <Final Arrival Time>

# TOTAL TIME: <Time from First Departure Time to Final Arrival Time>

Thank you for booking with LDF and we look forward to doing business with you again.

queries@ldf.ie www.lef.ie 021-123456 www.facebook.com/ldf

For example, the journey summary for the journey detailed on page 2 should be:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SUMMARY OF JOURNEY    Name: Mary Smith |  | CLASS: ECONOMY |  | COST: €1229.94 |
| FROM |  | DATE |  | TIME |
| ORK |  | 01.02.2020 |  | 10:45AM |
| TO  DXB    TOTAL TIME: 33H 30M |  | 02.02.2020 |  | 20:15PM |

The boarding passes and summary letter are then placed in an envelope and given to the customer.

At the end of the day, a report is generated and lists the number of flights sold that day, the total cost of the flights, and the commission earned by the travel agent (10% of the cost of the flights).

Design, code and test an Object-Oriented C++ program that will fulfil the needs of LDF Travel.

**SUBMIT**

**Include each of the following in 1 pdf document:**

1. The **design/algorithm** of your program.

This will include the preparatory work that you did before coding. It should contain at least:

* + The name of the object(s) that you intend coding.
  + The class(es) that the object(s) belong to.
  + The members (name and data type) that your class(es) will contain.
  + The methods (name, parameter list and/or return type) of any methods that you

intend to implement in your program.

* + An algorithm for each of the methods.
  + An algorithm for the main program.

Include any other documentation that you consider relevant.

1. Your **program code**. Ensure that the code is suitably commented. If you are using separate class and implementation files, ensure that you include all classes and implementation files in the document.

1. The **test data** that you developed.
   * You are required to run the program with the test data shown above and you are required to develop at least three other items of test data and expected results. In each case, specify exactly what you are testing.
   * Submit the screen shots that you took while testing the program.
   * Indicate whether your test data results match the actual results or not.

**Upload that document to Moodle.**

After uploading that document, upload your source code. If you have multiple files, you can either zip them and upload the zip file, or upload each file individually.

**Medical Cert/Extenuating Circumstances Form**

If you have been granted an extension because of medical circumstances or extenuating circumstances, then you must upload the relevant medical cert/extenuating circumstances form.

**6N2108 Assignment One Milan Labus**

**Design**

**Objects:**

1. Customer

2. Boarding Pass

**Classes:**

1. Person Class

2. Boarding pass Class

**Members:**

1. Person Class: firstName(string), surname(string), Address1 (string), Address2 (string), Address3 (string)

2. Boarding pass class: Departure Hour (Btye), Departure Minute (Byte), Arrival Hour (Byte), Arrival Minute (Byte), Departure Date (dd/MM/yyyy), Arrival Date (dd/MM/yyyy), First Airport1 (string), Second Airport (string), Third Airport (string) Flight Class (string), Flight Time1 (float), Flight Time 2 (float) , total Flight time (float), Cost1(float), Cost2(float) , Total Cost (float).

**Methods:**

Person Class:

* Void getCustomerDetails();

BoardingPassClass:

* Void getFlightDetails();
* Void calcFlightTime();
* Void printBoardingPass();
* Void printSummaryLetter();
* Void generateReport();

**Algorithm:**

**Person Class:**

**getCustomerDetails();**

OUTPUT “Please enter your customers name ”

INPUT firstName

OUTPUT “please enter the customers surname”

INPUT surname

OUTPUT “Please Enter Address Line 1”

INPUT Address1

OUTPUT “Please Enter Address Line 2”

INPUT Address2

OUTPUT “Please Enter Address Line 3”

INPUT Address3

**Boarding Pass Class:**

**getFlightDetails()**

OUTPUT “Select which flight class you would like to be seated in”

OUTPUT 1: First Class 2: Business 3: Premium Economy 4: Economy"

INPUT classChoice

Switch = classChoice

Case1 : flightClass=”First Class”

Case2 : flightClass=”Business Class”

Case3 : flightClass=”Premium Economy Class”

Case4 : flightClass = “Economy Class”

Default : “Error please enter one of the options above”

OUTPUT "Please enter which airport you wish to depart from for the first leg using the airports 3 digit code”

INPUT firstAirport

OUTPUT "Please enter which airport you wish to travel to in the first leg using the airports 3 digit code”

INPUT secondAirport

OUTPUT "Please Enter your first leg Departure Date in the format DD/mm/YYYY”

INPUT firstDepartureDay

IF cin.get != “/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT firstDepartureMonth

IF cin.get !=”/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT firstDepartureYear

END IF

OUTPUT "Please Enter your first Arrival Date for the first leg in the format DD/mm/YYYY"

INPUT firstArrivalDay

IF cin.get != “/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT firstArrivalMonth

IF cin.get !=”/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT firstArrivalYear

END IF

OUTPUT “Enter the first leg Departure Hour (24h format)”

INPUT firstDepatureHour

IF firstDeparutreHour>23

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” firstDepartureHour

END IF

OUTPUT “Enter the first leg DepartureMinute (24h format)”

INPUT firstDepartureMinute

IF firstDepartureMinute>59

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” firstDeparturelMinute

END IF

OUTPUT “Enter the first leg arrival Hour (24h format)”

INPUT firstArrivalHour

IF firstArrivalHour>23

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” firstArrivalHour

END IF

OUTPUT “Enter the first leg arrival Minute (24h format)”

INPUT firstArrivalMinute

IF firstArrivalMinute>59

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” firstArrivalMinute

END IF

OUTPUT “Please Enter the Cost of the first leg”

INPUT cost1

OUTPUT “Enter the final destination airport using the 3 digit code”

INPUT thirdAirport

OUTPUT “Please Enter the second leg Depature Date in the format DD/mm/YYYY”

INPUT secondDepartureDay

IF cin.get != “/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT secondDepartureMonth

IF cin.get !=”/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT secondDepartureYear

END IF

OUTPUT “Please Enter the second Arrival Date for the second leg in the format DD/mm/YYYY”

INPUT secondArrivalDay

IF cin.get != “/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT secondArrivalMonth

IF cin.get !=”/”

OUTPUT “Error please use the exact format DD/mm/YYYY”

ELSE

INPUT secondArrivalYear

END IF

OUTPUT “Enter the second leg Departure Hour (24h format)”

INPUT secondDepatureHour

IF secondDeparutreHour>23

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” secondDepartureHour

END IF

OUTPUT “Enter the second leg Departure Minute (24h format)”

INPUT secondDepatureMinute

IF secondDeparutreMinute>59

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” secondDepartureMinute

END IF

OUTPUT “Enter the second leg Arrival Hour (24h format)”

INPUT second ArrivalHour

IF second Arrival Hour>23

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” secondArrivalHour

END IF

OUTPUT “Enter the second leg Arrival Minute (24h format)”

INPUT secondArrivalMinute

IF secondArrivalMinute>59

OUTPUT ”wrong input please enter an hour in the 24h format”

ELSE

OUTPUT “You Have Entered” secondArrivalMinute

END IF

OUPUT “Please Enter the Cost of the Second Leg”

INPUT cost2

SET totalCost=cost1+cost2

**calcFlightTime()**

SET firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute

SET firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute

IF firstDepartureDay<firstArrivalDay

SET firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute

SET firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute

SET firstDifferenceToMidnight = (24\*60) - firstDepartureTime

SET firstLegFlightTime= firstDifferenceToMidnight + firstArrivalTime

SET firstLegFlightHours = firstLegFlightTime/60

SET firstLegFlightMinutes = firstLegFlightTime%60

OUTPUT “First Leg Travel Time is” firstLegFlightHours + “Hrs”

OUTPUT FirstLegFlightMinutes + “Mins”

ELSE

SET firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute

SET firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute

SET firstLegFlightTime = firstArrivalTime-firstDepartureTime

SET firstLegFlightHours = firstLegFlightTime/60

SET firstLegFlightMinutes = firstLegFlightTime%60

OUTPUT “First Leg Travel Time is” firstLegFlightHours + “Hrs”

OUTPUT FirstLegFlightMinutes + “Mins”

END IF

SET SecondDepartureTime = (SecondDepartureHour\*60) + SecondDepartureMinute

SET SecondArrivalTime = (SecondArrivalHour\*60) + SecondArrivalMinute

IF firstDepartureDay<firstArrivalDay

SET SecondDepartureTime = (SecondDepartureHour \*60) + SecondDepartureMinute

SET SecondArrivalTime = (SecondArrivalHour \*60) + SecondArrivalMinute

SET secondDifferenceToMidnight = (24\*60) - SecondDepartureMinute

SET secondLegFlightTime= secondDifferenceToMidnight + SecondArrivalMinute

SET secondLegFlightHours = secondLegFlightTime/60

SET secondLegFlightMinutes = secondLegFlightTime%60

OUTPUT “Second Leg Travel Time is” secondLegFlightHours + “Hrs”

OUTPUT secondLegFlightMinutes + “Mins”

ELSE

SET SecondDepartureTime = (SecondDepartureHour \*60) + SecondDepartureMinute

SET SecondArrivalTime = (SecondArrivalHour \*60) + SecondArrivalMinute

SET secondLegFlightTime = secondArrivalTime-secondDepartureTime

SET secondLegFlightHours = secondLegFlightTime/60

SET secondLegFlightMinutes = secondLegFlightTime%60

OUTPUT “Second Leg Travel Time is” secondLegFlightHours + “Hrs”

OUTPUT secondLegFlightMinutes + “Mins”

END IF

SET totalFlightMinutes= firstLegFlightMinutes + secondLegFlightMinutes

SET totalFlightHours = firstLegFlightHours + secondLegFlightHours + (totalFlightMinutes/60)

SET totalFlightMinutes = totalFlightMinutes%60

OUTPUT “Total Leg Travel Time is” totalFlightHours + “Hrs”

OUTPUT totalFlightMinutes + “Mins

**printBoardingPass()**

OUTPUT “BOARDING PASS: ” + firstName + surname

OUTPUT “CLASS: “ + flightClass

OUTPUT “COST: “ + cost1

OUTPUT “FROM” + “DATE” + “Time”

OUTPUT firstAirport + firstDepartureDay + firstDepartureMonth + firstDepartureYear + firstDepartureHour + firstDepartureMinute

OUTPUT “TO”

OUTPUT thirdAirport + secondArrivalDay + secondArrivalMonth + secondArrivalYear secondArrivalHour + secondArrivalMinute

OUTPUT “EXPECTED FLIGHT TIME”

OUTPUT secondLegFlightHours + secondLegFlightMinutes

**printSummaryLetter()**

OUTPUT “LDF TRAVEL”

OUTPUT “Name” + firstName + surname

OUTPUT “Address” + Address1 + Address2 + Address3

OUTPUT “REF” + firstAirport + thirdAirport + secondArrivalDay + secondArrivalMonth secondArrivalYear + secondArrivalHour + secondArrivalMinute

OUTPUT “EXPECTED FLIGHT TIME” + totalFlightHours + totalFlightMinutes

OUTPUT "Thank you for booking with LDF and we look forward to doing business with you again"

OUTPUT [queries@idf.ie \t www.lef.ie \t 021-123456 \t www.facebook.com/idf](mailto:queries@idf.ie%20\t%20www.lef.ie%20\t%20021-123456%20\t%20www.facebook.com/idf)

**generateReport()**

OUTPUT “Report”

OUTPUT “Total Flights= ” + totalFlights

OUTPUT “Total Cost=” + totalCost

OUTPUT “Commission” + totalCost/10

**Data Dictionary**

Person Class:

|  |  |
| --- | --- |
| **Variable** | **Data Type** |
| firstName | string |
| surName | string |
| Address | string |

Boarding Pass Class:

|  |  |
| --- | --- |
| **Variable** | **Data Type** |
| classChoice | int |
| flightClass | string |
| DepartureAiportOne | string |
| DepartureAirportTwo | string |
| ArrivalAirportOne | string |
| ArrivalAirportTwo | string |
| DepartureHourOne | Byte |
| DepartureMinuteOne | Byte |
| DepartureHourTwo | Byte |
| DepartureMinuteTwo | Byte |
| ArrivalHourOne | Byte |
| ArrivalMinuteOne | Byte |
| ArrivalHourTwo | Byte |
| ArrivalMinuteTwo | Byte |
| TravelClass | string |
| FlightHourOne | Byte |
| FlightMinuteOne | Byte |
| FlightHourTwo | Byte |
| FlightMinuteTwo | Byte |
| TotalFlightHourOne | Byte |
| TotalFlightMinuteOne | Byte |
| TotalFlightHourTwo | Byte |
| TotalFlightMinuteTwo | Byte |
| FinalFlightHour | Byte |
| FinalFlightMinute | Byte |
| FlightCost1 | float |
| FlightCost2 | float |
| FinalCost | float |
| totalFlights | int |

**TESTING**

**Test1:**

Expected Results: We want to see if it works correctly when the entire journey is the same date in time

Flight1

Flight Departure 1

Date: 29/01/2021 Time: 01:00

Flight Arrival 1

Date: 29/01/2021 Time: 05:00

Cost1 : 400

First Leg Time: 4Hrs 0Mins

Flight2

Flight Departure 2

Date: 29/01/2021 Time: 06:00

Flight Arrival 2

Date: 29/01/2021 Time: 11:00

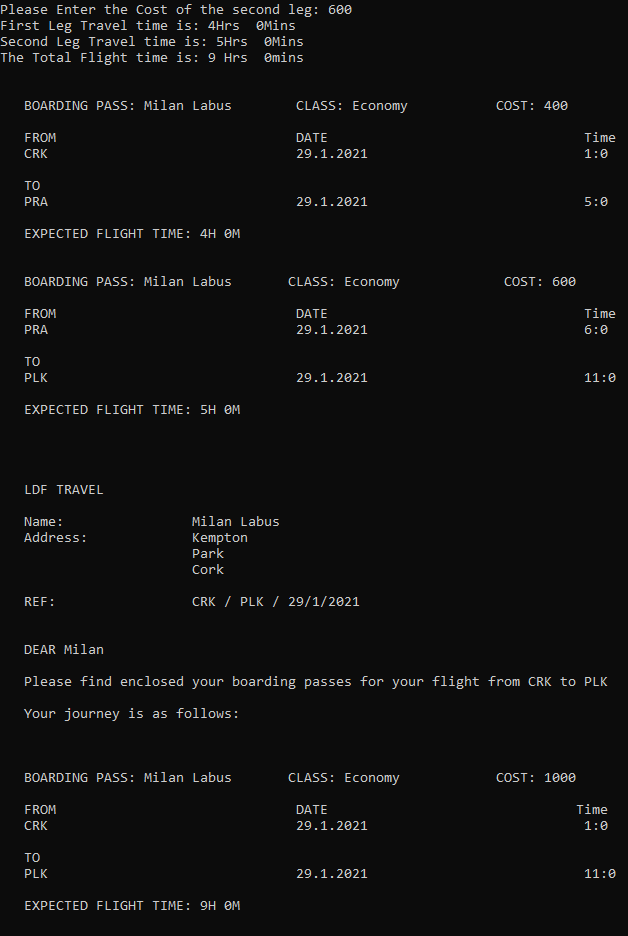
Cost1 : 600

Second Leg Time: 5Hrs 0Mins

Total Flight Time: 9Hrs 0Mins

Total Cost: 1000

**Actual Result:**



The program worked successfully.

**Test 2:**

Expected Results:

Flight1

Flight Departure 1

Date: 29/01/2021 Time: 22:30

Flight Arrival 1

Date: 30/01/2021 Time: 02:10

Cost1 : 550

First Leg Time: 3Hrs 40Mins

Flight2

Flight Departure 2

Date: 30/01/2021 Time: 04:00

Flight Arrival 2

Date: 30/01/2021 Time: 14:30

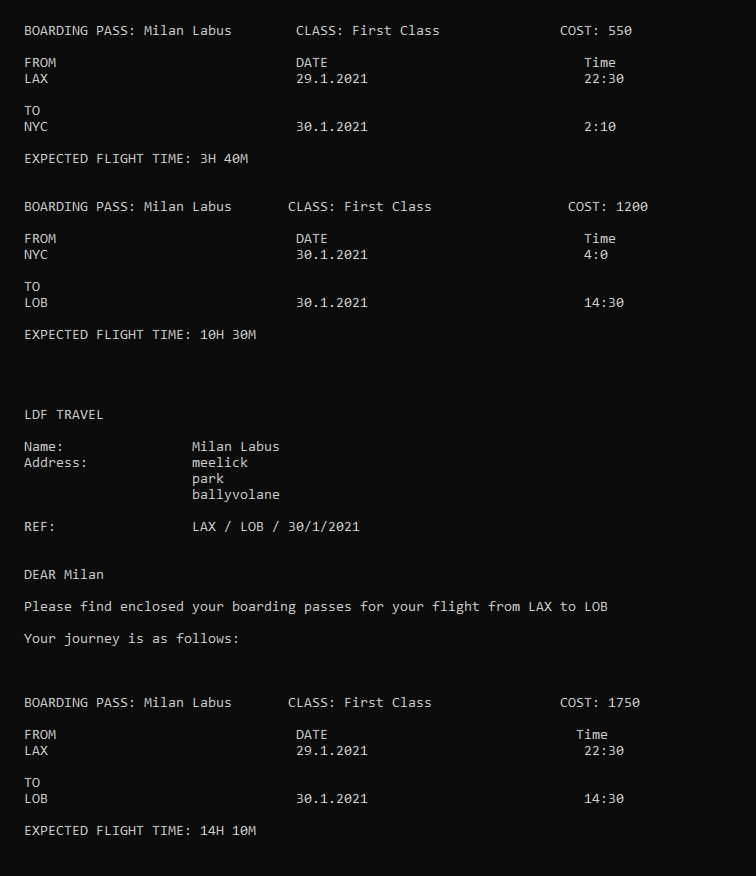
Cost1 : 1200

Second Leg Time: 10Hrs 30Mins

Total Flight Time: 14Hrs 10Mins

Total Cost: 1750

Actual Results:



This test was successful the program output the expected result

**Test 3:**

Expected Results: We will test if it works when the date and month changes during the journey

Flight1

Flight Departure 1

Date: 31/01/2021 Time: 20:00

Flight Arrival 1

Date: 01/02/2021 Time: 11:00

Cost1 : 2000

First Leg Time: 15Hrs 0Mins

Flight2

Flight Departure 2

Date: 01/02/2021 Time: 12:00

Flight Arrival 2

Date: 01/02/2021 Time: 23:00

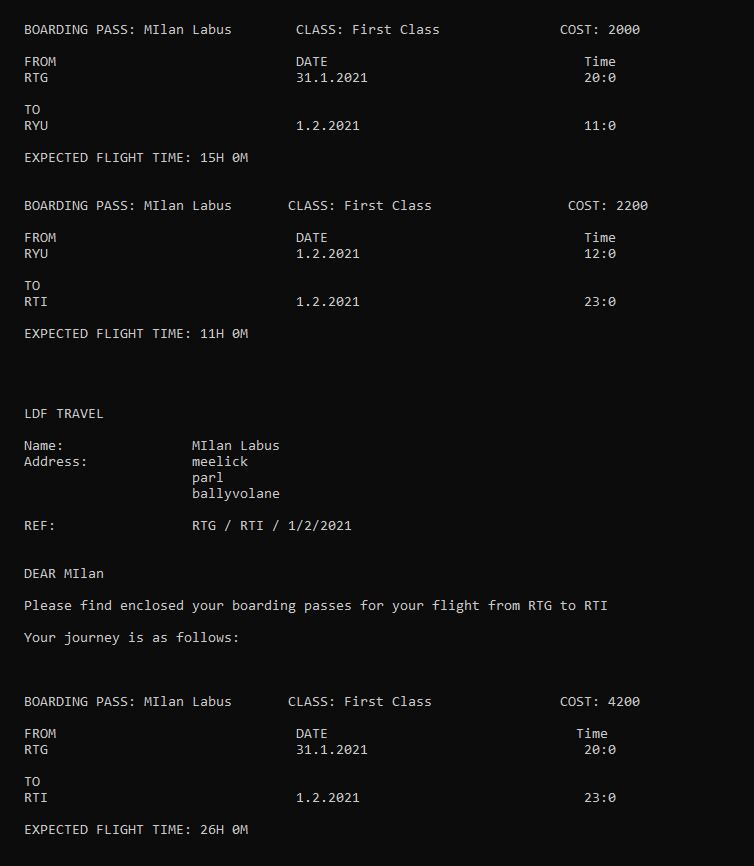
Cost1 : 2200

Second Leg Time: 11Hrs 00Mins

Total Flight Time: 26Hrs 0Mins

Total cost: 4200

Actual Result:



The Program works perfectly even if the month changes

Source Code

**Main**

#include "BoardingPass.h"

#include "Person.h"

#include <iostream>

using namespace std;

int main()

{

char again = 'Y';

while (again == 'y' || again == 'Y') //while loop to run program again

{

Person person;

BoardingPass ticketOne,ticketTwo;

person.getCustomerDetails();

ticketOne.getFlightDetails();

ticketOne.calcFlightTime();

ticketOne.printBoardingPass();

ticketOne.printSummaryLetter();

ticketOne.generateReport();

cout << "Would you like to run again? (y/n):";

cin >> again;//change control variable

} //end while loop

cout << "The end." << endl;

return 0;

}

**BoardingPass Class.h**

#ifndef BOARDINGPASS\_H

#define BOARDINGPASS\_H

#include "Person.h"

#include <string>

using namespace std;

class BoardingPass:public Person

{

public:

void getFlightDetails();

void calcFlightTime();

void printBoardingPass();

void printSummaryLetter();

void generateReport();

private:

int classChoice;

string flightClass;

//calculating cost using a multiplier

float cost1;

float cost2;

float totalCost;

int totalFlights;

//first leg of the journey

int firstDepartureDay;

int firstDepartureMonth;

int firstDepartureYear;

int firstArrivalDay;

int firstArrivalMonth;

int firstArrivalYear;

string firstAirport;

string secondAirport;

int firstDepartureHour;

int firstDepartureMinute;

int firstArrivalHour;

int firstArrivalMinute;

//second leg of the journeyint firstDepartureDay;

int secondDepartureDay;

int secondDepartureMonth;

int secondDepartureYear;

int secondArrivalDay;

int secondArrivalMonth;

int secondArrivalYear;

int secondDepartureHour;

int secondDepartureMinute;

string thirdAirport;

int secondArrivalHour;

int secondArrivalMinute;

//Flight Time for the first flight

int firstDepartureTime;

int firstArrivalTime;

int firstDifferenceToMidnight;

int firstLegFlightTime;

int firstLegFlightHours;

int firstLegFlightMinutes;

//Flight Time for the second flight

int secondDepartureTime;

int secondArrivalTime;

int secondDifferenceToMidnight;

int secondLegFlightTime;

int secondLegFlightHours;

int secondLegFlightMinutes;

//Total Flight Time

int totalFlightHours;

int totalFlightMinutes;

};

#endif // BOARDINGPASS\_H

**BoardingPass.cpp**

#include "BoardingPass.h"

#include "Person.h"

#include <iostream>

#include <string>

using namespace std;

void BoardingPass::getFlightDetails()

{

// Now comes the seating class selection

cout << "\nPlease select which class you would like to be seated in: \n";

cout << " 1: First Class \n" << " 2: Business \n" << " 3: Premium Economy\n" << " 4: Economy";

cin >> classChoice;

switch (classChoice)

{

case 1:

flightClass = "First Class";

cout << "You have selected First Class" << "\n";

break;

case 2:

flightClass = "Business Class";

cout << "You have selected Business Class" << "\n";

break;

case 3:

flightClass = "Premium Economy";

cout << "You have selected Premium Economy Class" << "\n";

break;

case 4:

flightClass = "Economy";

cout << "You have selected Economy Class" << "\n";

break;

default:

cout << "Error please enter one of the options above \n";

}

//details for the first leg

cout << "Please enter which airport you wish to depart from for the first leg using the airports 3 digit code: \n";

cin >> firstAirport;

cout << "Please enter which airport you wish to travel to in the first leg using the airports 3 digit code: \n";

cin >> secondAirport;

//promt the user to enter the first leg departure date in the correct format

cout << "Please Enter your first leg Departure Date in the format DD/mm/YYYY: ";

cin >> firstDepartureDay; // read the day

if ( cin.get() != '/' ) // make sure there is a slash between DD and MM

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> firstDepartureMonth; // read the month

if ( cin.get() != '/' ) // make sure there is a slash between MM and YYYY

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> firstDepartureYear; // read the year

cout << "chosen deparutre date: " << firstDepartureDay << "/" << firstDepartureMonth << "/" << firstDepartureYear << "\n";

//prompt the user to enter the arrival date for the first leg in the correct format

cout << "Please Enter your first Arrival Date for the first leg in the format DD/mm/YYYY: ";

cin >> firstArrivalDay;

if ( cin.get() != '/' ) // make sure there is a slash between DD and MM

{

cout << "Error please use the exact format DD/mm/YYYY \n";

}

cin >> firstArrivalMonth;

if ( cin.get() != '/' )

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> firstArrivalYear; // read the year

cout << "chosen arrival date: " << firstArrivalDay << "/" << firstArrivalMonth << "/" << firstArrivalYear << "\n";

//Now we will get the user to enter their departure time

cout << "Enter the first leg Departure Hour (24h format): " << "\n";

cin >> firstDepartureHour;

if(firstDepartureHour>23)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << firstDepartureHour << "\n";

}

cout << "Enter the first leg Departure Minute (24h format): " << "\n";

cin >> firstDepartureMinute;

if(firstDepartureMinute>59)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << firstDepartureMinute << "\n";

}

//Now we will get the user to enter their firstArrival time

cout << "Enter the first leg Arrival Hour (24h format): " << "\n";

cin >> firstArrivalHour;

if(firstArrivalHour>23)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << firstArrivalHour << "\n";

}

cout << "Enter the first leg Arrival Minute (24h format): " << "\n";

cin >> firstArrivalMinute;

if(firstArrivalMinute>59)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << firstArrivalMinute << "\n";

}

cout << "Please Enter the Cost of the first leg: ";

cin >> cost1;

//now for the second leg details

//Now to get which airport they want to depart from for the first leg of the journey

cout << "Enter the final destination airport using the 3 digit code";

cin >> thirdAirport;

//promt the user to enter the second leg departure date in the correct format

cout << "Please Enter the second leg Depature Date in the format DD/mm/YYYY: ";

cin >> secondDepartureDay; // read the day

if ( cin.get() != '/' ) // make sure there is a slash between DD and MM

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> secondDepartureMonth; // read the month

if ( cin.get() != '/' ) // make sure there is a slash between MM and YYYY

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> secondDepartureYear; // read the year

cout << "chosen deparutre date: " << secondDepartureDay << "/" << secondDepartureMonth << "/" << secondDepartureYear << "\n";

//prompt the user to enter the arrival date for the second leg in the correct format

cout << "Please Enter the second Arrival Date for the second leg in the format DD/mm/YYYY: ";

cin >> secondArrivalDay;

if ( cin.get() != '/' ) // make sure there is a slash between DD and MM

{

cout << "Error please use the exact format DD/mm/YYYY \n";

}

cin >> secondArrivalMonth;

if ( cin.get() != '/' )

{

cout << "Error please use the exact format DD/mm/YYYY\n";

}

cin >> secondArrivalYear; // read the year

cout << "chosen arrival date: " << secondArrivalDay << "/" << secondArrivalMonth << "/" << secondArrivalYear << "\n";

//Now we will get the user to enter their second departure time

cout << "Enter the Second leg Departure Hour (24h format): " << "\n";

cin >> secondDepartureHour;

if(secondDepartureHour>23)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << secondDepartureHour << "\n";

}

cout << "Enter the second leg Departure Minute (24h format): " << "\n";

cin >> secondDepartureMinute;

if(secondDepartureMinute>59)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << secondDepartureMinute << "\n";

}

//Now we will get the user to enter their second Arrival time

cout << "Enter the second leg Arrival Hour (24h format): " << "\n";

cin >> secondArrivalHour;

if(secondArrivalHour>23)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << secondArrivalHour << "\n";

}

cout << "Enter the first leg Arrival Minute (24h format): " << "\n";

cin >> secondArrivalMinute;

if(secondArrivalMinute>59)

{

cout << "wrong input please enter an hour in the 24h format";

}

else

{

cout << "You have Entered: " << secondArrivalMinute << "\n";

}

cout << "Please Enter the Cost of the second leg: ";

cin >> cost2;

totalCost=cost1+cost2;

}

void BoardingPass::calcFlightTime()

{

//Calculating the first leg flight time

//converting the first the times to minutes

firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute;

firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute;

//if the first day,month or year is different to the second day i will find the difference to midnight

if(firstDepartureDay<firstArrivalDay || firstDepartureMonth<firstArrivalMonth || firstDepartureYear<firstArrivalYear)

{

firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute;

firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute;

firstDifferenceToMidnight = (24\*60) - firstDepartureTime;

firstLegFlightTime= firstDifferenceToMidnight + firstArrivalTime;

firstLegFlightHours = firstLegFlightTime/60;

firstLegFlightMinutes = firstLegFlightTime%60;

cout << "First Leg Travel time is: " << firstLegFlightHours << "Hrs" << " " << firstLegFlightMinutes << "Mins" << "\n";

}

//if the flight is on the same day we dont need to find the difference to midnight

else

{

firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute;

firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute;

firstLegFlightTime = firstArrivalTime-firstDepartureTime;

firstLegFlightHours = firstLegFlightTime/60;

firstLegFlightMinutes = firstLegFlightTime%60;

cout << "First Leg Travel time is: " << firstLegFlightHours << "Hrs" << " " << firstLegFlightMinutes << "Mins" << "\n";

}

firstDepartureTime = (firstDepartureHour\*60) + firstDepartureMinute;

firstArrivalTime = (firstArrivalHour\*60) + firstArrivalMinute;

//now to do the same for the second leg of the joruney

//Calculating the first leg flight time

//converting the first the times to minutes

secondDepartureTime = (secondDepartureHour\*60) + secondDepartureMinute;

secondArrivalTime = (secondArrivalHour\*60) + secondArrivalMinute;

//if the first day,month or year is different to the second day i will find the difference to midnight

if(secondDepartureDay<secondArrivalDay || secondDepartureMonth<secondArrivalMonth || secondDepartureYear<secondArrivalYear)

{

secondDepartureTime = (secondDepartureHour\*60) + secondDepartureMinute;

secondArrivalTime = (secondArrivalHour\*60) + secondArrivalMinute;

secondDifferenceToMidnight = (24\*60) - secondDepartureTime;

secondLegFlightTime= secondDifferenceToMidnight + secondArrivalTime;

secondLegFlightHours = secondLegFlightTime/60;

secondLegFlightMinutes = secondLegFlightTime%60;

cout << "Second Leg Travel time is: " << secondLegFlightHours << "Hrs" << " " << secondLegFlightMinutes << "Mins" << "\n";

}

//if the flight is on the same day we dont need to find the difference to midnight

else

{

secondDepartureTime = (secondDepartureHour\*60) + secondDepartureMinute;

secondArrivalTime = (secondArrivalHour\*60) + secondArrivalMinute;

secondLegFlightTime = secondArrivalTime-secondDepartureTime;

secondLegFlightHours = secondLegFlightTime/60;

secondLegFlightMinutes = secondLegFlightTime%60;

cout << "Second Leg Travel time is: " << secondLegFlightHours << "Hrs" << " " << secondLegFlightMinutes << "Mins" << "\n";

}

//now to calculate total flight time

totalFlightMinutes= firstLegFlightMinutes + secondLegFlightMinutes;

totalFlightHours = firstLegFlightHours + secondLegFlightHours + (totalFlightMinutes/60);

totalFlightMinutes = totalFlightMinutes%60;

cout << "The Total Flight time is: " << totalFlightHours << " Hrs" << " " <<totalFlightMinutes << "mins" << "\n\n\n";

}

void BoardingPass::printBoardingPass()

{

cout << " BOARDING PASS: " << firstName << " " << surName << " " << " \tCLASS: " << flightClass << " \t\t" << "\tCOST: " << cost1 << " \n\n";

cout << " FROM" << "\t\t\t\t DATE" << "\t\t\t Time \n";

cout << " " << firstAirport << "\t\t\t\t " << firstDepartureDay << "." << firstDepartureMonth << "." << firstDepartureYear << "\t\t\t " << firstDepartureHour << ":" << firstDepartureMinute << "\t\n\n";

cout << " TO" << "\n";

cout << " " << secondAirport << "\t\t\t\t " << firstArrivalDay << "." << firstArrivalMonth << "." << firstArrivalYear << "\t\t\t " << firstArrivalHour << ":" << firstArrivalMinute << "\t\n\n";

cout << " " << "EXPECTED FLIGHT TIME: " << firstLegFlightHours << "H " << firstLegFlightMinutes << "M" << "\n\n\n";

totalFlights++;

cout << " BOARDING PASS: " << firstName << " " << surName << " \t" << " CLASS: " << flightClass << " \t\t" << "\tCOST: " << cost2 << " \n\n";

cout << " FROM" << "\t\t\t\t DATE" << "\t\t Time \n";

cout << " " << secondAirport << "\t\t\t\t " << secondDepartureDay << "." << secondDepartureMonth << "." << secondDepartureYear << "\t\t\t " << secondDepartureHour << ":" << secondDepartureMinute << "\t\n\n";

cout << " TO" << "\n";

cout << " " << thirdAirport << "\t\t\t\t " << secondArrivalDay << "." << secondArrivalMonth << "." << secondArrivalYear << "\t\t\t " << secondArrivalHour << ":" << secondArrivalMinute << "\t\n\n";

cout << " " << "EXPECTED FLIGHT TIME: " << secondLegFlightHours << "H " << secondLegFlightMinutes << "M" << "\n\n\n\n\n";

totalFlights++;

}

void BoardingPass::printSummaryLetter()

{

//printing the summary letter

cout << " LDF TRAVEL\n\n";

cout << " Name: \t\t" << firstName << " " << surName << "\n";

cout << " Address:\t\t" << Address1 << "\n";

cout << "\t\t\t" << Address2 << "\n";

cout << "\t\t\t" << Address3 << "\n\n";

cout << " REF: \t\t" << firstAirport << " / " << thirdAirport << " / " << secondArrivalDay << "/" << secondArrivalMonth << "/" << secondArrivalYear << "\n\n\n";

cout << " DEAR " << firstName << "\n\n";

cout << " Please find enclosed your boarding passes for your flight from " << firstAirport << " to " << thirdAirport << " " << "\n\n";

cout << " Your journey is as follows: \n\n\n\n";

cout << " BOARDING PASS: " << firstName << " " << surName << " " << " \t CLASS: " << flightClass << " \t" << "\t\tCOST: " << totalCost << " \n\n";

cout << " FROM" << "\t\t\t\t DATE" << "\t\t Time \n";

cout << " " << firstAirport << "\t\t\t\t " << firstDepartureDay << "." << firstDepartureMonth << "." << firstDepartureYear << "\t\t\t " << firstDepartureHour << ":" << firstDepartureMinute << "\t\n\n";

cout << " TO" << "\n";

cout << " " << thirdAirport << "\t\t\t\t " << secondArrivalDay << "." << secondArrivalMonth << "." << secondArrivalYear << "\t\t\t " << secondArrivalHour << ":" << secondArrivalMinute << "\t\n\n";

cout << " " << "EXPECTED FLIGHT TIME: " << totalFlightHours << "H " << totalFlightMinutes << "M" << "\n\n\n\n";

cout << " Thank you for booking with LDF and we look forward to doing business with you again." << "\n\n";

cout << " queries@idf.ie \t www.lef.ie \t 021-123456 \t www.facebook.com/idf" << "\n\n\n\n\n";

}

void BoardingPass::generateReport()

{

//generating report

cout << " Report: \n\n\n";

cout << " Total Flights = " << totalFlights << "\n";

cout << " Total cost = " << totalCost << "\n";

cout << " Commission = " << totalCost/10 << "\n\n\n";

}

**Person.h**

#ifndef PERSON\_H

#define PERSON\_H

#include <string>

using namespace std;

class Person

{

public:

void getCustomerDetails();

protected:

string firstName;

string surName;

string Address1;

string Address2;

string Address3;

};

#endif // PERSON\_H

**Person.cpp**

#include "Person.h"

#include <iostream>

#include <string>

using namespace std;

void Person::getCustomerDetails()

{

cout << "please enter your customers name: ";

cin >> firstName;

cout << "please enter the customers surname: ";

cin >> surName;

cout << "You have Entered the name: " << firstName << " " << surName << endl;

cout << "Please Enter Address Line 1: ";

cin >> Address1;

cout << "Please Enter Address Line 2: " ;

cin >> Address2;

cout << "Please Enter Address Line 3: " ;

cin >> Address3;

}