Remas Bashanfar

UID: 505597737

CS31

Project 5

Obstacles:

I struggled with the last Codeboard test as my balance did not add up to zero. However, after many tries and TA Office Hours, I was able to overcome this problem and realized that I should not subtract mileage from balance in “can earn free flight” but only in “free flight” since the person might be able to get a free flight but may not take it. While for freeflight, the person takes the free flight so we have to subtract mileage from balance here.

Some Test Cases:

To elaborate on the point about the test case that I had a problem with, this is the test case that was going wrong but it worked in the end.

PlaneFlight f( "Howard", "LAX", "LAS", 65.00, 285 );

FrequentFlyerAccount account( "Howard" );

assert( account.getName() == "Howard" );

assert( std::to\_string( account.getBalance() ) == "285.000000" );

account.addFlightToAccount( f );assert( account.canEarnFreeFlight( 285 ) );account.freeFlight( "LAS", "LAX", 285, f );

assert( f.getName() == "Howard" );

assert( f.getFromCity() == "LAS" );

assert( f.getToCity() == "LAX" );

assert( f.getCost() == 0 );

assert( f.getMileage() == 285 );

account.addFlightToAccount( f );

assert( account.getBalance( ) == 0 );

For the assertions that my program handled well:

Assertion to make sure the gettocity, getfromcity, getmileage and getname works well:

PlaneFlight shortleg( "Howard", "LAX", "LAS", 49.00, 285 );

PlaneFlight longleg( "Howard", "LAS", "NYC", 399.00, 2800 );

PlaneFlight sample( "Sample", "Sample", "Sample", 0, 1 );

FrequentFlyerAccount account( "Howard" );

assert( shortleg.getFromCity( ) == "LAX" );

assert( shortleg.getToCity( ) == "LAS" );

assert( shortleg.getName( ) == "Howard" );

assert( std::to\_string( shortleg.getCost( ) ) == "49.000000" );

assert( std::to\_string( shortleg.getMileage( ) ) == "285.000000" );

Assertion to make sure the get balance and can earn free flight works well:

assert( std::to\_string( account.getBalance( ) ) == "0.000000" );

assert( account.getName( ) == "Howard" );

assert( account.canEarnFreeFlight( 3300.00 ) == **false** );

Assertion to make sure add flight and get balance works well:

assert( account.addFlightToAccount( shortleg ) == **true** ); // returns true because the names match

assert( account.addFlightToAccount( longleg ) == **true** ); // returns true because the names match

assert( std::to\_string( account.getBalance( ) ) == "3085.000000" );

Assertion to make sure free flight, get mileage, get from city, get to city, get cost works well:

**if** (account.canEarnFreeFlight( 285 ))

{

assert( account.freeFlight( "LAS", "LAX", 285, sample ) == **true** );

// Howard earned a free flight...

assert( sample.getName( ) == "Howard" );

assert( std::to\_string( sample.getCost( ) ) == "0.000000" );

assert( sample.getFromCity( ) == "LAS" );

assert( sample.getToCity( ) == "LAX" );

assert( std::to\_string( sample.getMileage( ) ) == "285.000000" );

// account has been reduced for this free flight...

assert( std::to\_string( account.getBalance( ) ) == "2800.000000" );

}

**else**

{

assert( **false** ); // there are enough miles in the account...

}

Assertion to make sure add flight to account and get balance works well:

PlaneFlight muffin( "Muffin", "LAX", "Doggie Heaven", 500, 500 );

assert( account.addFlightToAccount( muffin ) == **false** );

assert( std::to\_string( account.getBalance( ) ) == "2800.000000" );

Here is a list of other assertions that worked that I got from codeboard:

Passed test case-check all flight getters:

Input:

flight 1

Expected output was correct: Howard LAX LAS 65 285

Passed test case-check all flight setters:

Input:

flight 2

Expected output was correct: Pixie CHI BOS 100 100

Passed test case-name cannot be empty:

Input:

flight 3

Expected output was correct: Howard

Passed test case-fromCity cannot match toCity:

Input:

flight 4

Expected output was correct: LAX LAS

Passed test case-mileage cannot be 0 or less:

Input:

flight 5

Expected output was correct: -1

Passed test case-cost can be zero:

Input:

flight 6

Expected output was correct: 0

Passed test case-cost cannot be less than 0:

Input:

flight 7

Expected output was correct: -1

Passed test case-check all getters:

Input:

account 1

Expected output was correct: Howard 0.000000

Passed test case-cannot earn free flight with balance of zero:

Input:

account 2

Expected output was correct: false

Passed test case-cannot earn miles on flight with non-matching name:

Input:

account 3

Expected output was correct: Howard 0.000000

Passed test case-flight updates balance:

Input:

account 4

Expected output was correct: Howard 285.000000

Passed test case-earned flight reduces balance:

Input:

account 5

Expected output was correct: true Howard LAS LAX 0 285

Passed test case-cannot earn free flight if distance is too far:

Input:

account 6

Expected output was correct: false

Passed test case-free flight don't earn miles in flyer account:

Input:

account 7

Expected output was correct: Howard 0.000000

Score: 100. You passed 14 out of 14 cases