CPSC-121
Individual Project #3

The purpose of this assignment is to get experience with class and member functions.

The project requirements:

It is an important part of your grade that you design your application according to the following requirements.

Step 1. Create the UML class diagram. Refer to a sample posted.

Step 2. Create a pseudo code to list all of the actions to execute along with the order these actions should occur. Include your pseudo code on the top of your cpp file in the comment section after your name and Project# along with code comments.

Step 3. Reminder, it is important to document all your functions by including comments above all your functions prototypes in the class specification file. Describe the function's preconditions and postconditions. Precondition: A statement specifying the condition(s) that must be true before the function is called. Postcondition: A statement specifying what is true after the function call is completed.

Step 4. Write the source code according to your pseudo code, the UML diagram and the following project requirements. You must use a class and class member functions in your program. Keep in mind that it is considered good design to have class member (***accessors*** and ***mutators***) functions *avoid using cin and cout*. As a general rule, classes should provide ***accessor*** member functions for retrieving data values without displaying them on the screen. Likewise, they should provide ***mutator*** member functions that store data into private member variables without using cin. This allows a programmer to use the class without being locked into a particular method of performing I/O.

Step 5. Create 3 separate files as follows below.

- Place class declaration in a header file that serves as the class specification file. You can name the file **Airplane**.h
- Place member function definitions in a class implementation file. Name the file **Airplane**.cpp. This file should #include the class specification file (**Airplane**.h).
- A client program (client code, main.cpp) that uses the class must #include the class specification file and be compiled and linked with the class implementation file.

Step 6. Code a class with data members according your UML diagram. Your class should have the following member variables:

- year. An int that holds the airplane's year.
- model (and number). A string object that holds the model of the airplane. (example Boeing 737)
- speed. An int that holds the airplane's current speed.

• capacity. An int that holds the airplane's number of seats.

Step 7. Your class should have the following member functions:

- Constructor. The constructor should accept custom inputs from the user and set the airplane's year/model/capacity as arguments and assign these values to the object's attributes. The constructor should initialize the speed member variable to 0.
- Getter and setter (***accessors*** and ***mutators***) functions that allow values to be set or retrieved from all class attributes.
- Accelerate. The accelerate function should add 100 to the speed member variable each time it is called.
- Brake. The brake function should subtract 100 from the speed member variable each time it is called.
- Destructor. The destructor needs to display message "Thank you for flying!"

Step 8. Demonstrate the class in a program that creates an Airplane object based on custom user inputs, and then call the accelerate function five times (in a loop). After each call to the accelerate function, get the current speed of the airplane and display it.
Then, call the brake function five times (in a loop). After each call to the brake function, get the current speed of the airplane and display it.

Projects Submission Policy:

Your electronic file (PDF) must be submitted by the deadline listed on Titanium. The PDF should include your name, project#, code comments, UML diagram, C++ code and a screenshots of your program output. You should be able to explain any part of your program. The feedback on your program will be given in person during the lab hours. The grade of the project will be based on the following rule:

Program works correctly: 5

Program is mostly correct and has minor problem: 4

Program has multiple significant errors (combination of incorrect output or calculation errors or missed project requirements): 3

The following kinds of submissions cannot be evaluated, and will be assigned a zero score:

Email or late submissions.

Source code that cannot be compiled successfully.

Input/output that is falsified or does not match the submitted source code.

Submissions that are plagiarized.