Lab 5: Object-Oriented Analysis and Design

CPSC 1021

Lab Objective

- Practice object-oriented analysis and design
- Practice writing UML class diagrams
- Practice object-based programming
- Practice collaborating on a programming project

Introduction

This week you will start working with your group on a larger project. The goal of this project is to practice object-oriented analysis and design and to write an object-based program. Before you start on this project, make sure you and your team members have read Chapter 7.15 "Introduction to Object-Oriented Analysis and Design."

Collaboration Policy

Work with your assigned team members from Lab 0.

You and your team members can work on the analysis and design part of the project together. You can also collaborate on the main program. But each student should individually work on the classes they have been assigned (see further details below).

You and your partners can work on the assignments together.

Submission Policy

Code solutions to all challenges **can be submitted as teams** this week. **This means only one member from each team SHOULD to submit the code files on Gradescope** (this person must then add all their other team members to the submission). However, **you are required to work on all problems together**.

Description of the Problem Domain

Clemson University's Parking and Transportation Services is responsible for selling parking permits to employees, students, visitors, and vendors. Currently, customers can select from different permit types, such as annual permits, semester permits, one-day permits, park-and-ride permits, etc. Different permits are also available for different types of vehicles, such as regular cars, low-emission vehicles, motorcycles and motor scooters, etc.

Parking and Transportation Services has hired your team to revise its existing permit system and to develop a program that customers can use to purchase a permit under the new system.

Your program must allow customers to:

- provide their personal information (e.g., name, address, etc.)
- select a vehicle type (e.g., car, motorcycle, etc.)
- provide vehicle-related information (e.g., make, model, license plate, etc.)
- select a permit type (e.g., annual, semester, etc.)
- receive an invoice that includes the price for the permit and a summary of the information they have entered

You are free to decide who is required to buy a permit, what types of vehicles can be selected, what types of permits are offered, and the price of each permit.

Class Requirements

Your program must use the following classes, where *N* denotes the number of members in your team:

- At least *N* classes for *N* different types of customers. These classes must store name, address, email, and at least two class-specific attributes.
- At least *N* classes for *N* different types of vehicles. These classes must store make, model, year, and at least two class-specific attributes.
- A class named 'Invoice' which stores the price for the selected permit, at least one type of discount that you decide (cheaper permits for students or low-emission vehicles? free permits for CPSC 1020 professors and TAs?), and at least one service charge you decide. This class should include a member function that calculates the total price, and a member function that generates the invoice as a screen print-out that includes the total price and all information provided by the user.

Team Requirements

Decide as a team what your final permit system will look like and how you will design and test your program. After Lab 5, "Design & Test Plan", each team member must be responsible for implementing one customer type class and one vehicle type class. In your final submission, clearly indicate who has worked on which part by adding the team member's name as a comment on the top of the .h and .cpp file.

Milestones

The project is divided into three steps that should be completed by the following deadlines:

Step 1 – Analysis, Design & Test Plan

By the end of the first half of the first lab, you should have:

- Identified the classes and objects you need for your project
- Defined each class's attributes
- Defined each class's behavior
- Written the UML class diagram for each class
- Written pseudocode for your client program
- Finished a test plan that you will use in Lab 5 to check your final program for correctness
- Split up the work between team members. Each team member must write code for at least one customer type class and one vehicle type class

Step 2 – Implementation

By the end of the second half of the first lab, you should have:

- Completed the code for each class. This means each team member should have finished writing a class declaration (.h) and a class definition (.cpp) for each of the classes they were assigned.
- Written the Invoice class
- Written additional functions and code needed for your client program

Lab 3 – Integration & Testing

By the end of the week, you should have:

- Completed your client program, which means integrating all classes and functions into a single program
- Tested your program with the test plan you have completed in Lab 5
- Fixed bugs and errors revealed by your test plan

The following are the files you should submit to Gradescope.

Customer.h
Customer.cpp
Vehicle.h
Vehicle.cpp
Invoice.h
Invoice.cpp
main.cpp
makefile
inputTest1.txt
inputTest2.txt
inputTest3.txt
outputTest1.txt
outputTest2.txt
outputTest3.txt

Your input and output files will undergo testing in Gradescope along with other submissions, so it's essential to upload them accordingly. These files are available in the modules section, along with the lab manual.Basically these files contains how you will give your input and how your output should look like. Please use them as references to create your own input and output files rather than submitting them directly.