Learning and Reflection

1. **Explain** how you implemented the property structure and how you passed it to have the administrator enter the ride share information. Include how you dereference a pointer to a structure.

In the ride share project, I implemented a struct names rideshare to store information about the ride share service. To collect input from the administrator I passed a pointer to the struct to the setUpRideShare function. Within the function, I used the arrow operator to access the struct members and modify them.

Ex: **void** setUpRideShare (rideShare \*rideSharePtr){

printf("Set up rideshare information\n");

printf("Enter the RideShare Organization Name:\n ");

fgets(rideSharePtr -> organizationName, STRING\_LENGTH, stdin);

fgetsRemoveNewLine(rideSharePtr -> organizationName);

printf("Enter the base fare: ");

rideSharePtr -> baseFare = getValidDouble(MIN\_SET\_UP, MAX\_SET\_UP);

1. **Explain** what it means to implement maintainable and extendable code. Include at least 2 different examples from your code.

Implementing maintainable and extendable code means writing code that is easy to understand, modify, and build upon in the future. In the ride share project, I achieved maintainability and extensibility by using functions to encapsulate specific tasks, and defining constants.

1. Think about where you started at the beginning of the semester and where you are now with designing and developing a software solution. Share two of your biggest accomplishments so far this semester.

Two of my biggest accomplishments is being able to successfully implement user authentication being able to demonstrate my ability to handle security-related aspects of software development. Also being able to develop a structured approach to problem solving and code organization.

1. Share two strengths of yours for iteration 02 solution and at least one area you would like to improve on for iteration 03.

Two strengths are organizing my code into modular functions making it easier to maintain and incorporating input validation techniques.

One area I would like to improve for my iteration 03 is being able to enhance my error handling mechanisms

Test CASES

A close-up of a test

Description automatically generated

A close-up of a paper

Description automatically generated

A close-up of a paper

Description automatically generated

A close-up of a list of words

Description automatically generated