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 #2		
Issue Date:	8th March 2021	**Submission Date:**	25th March 2021
Learning Outcomes Assessed:	LO2, LO4, LO5, LO6, LO8		

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:	

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.

CALLOUT DISPATCH SERVICES

CallOut Dispatch Services has five taxis, two buses and three minibuses. The registration number, make, model, kilometres driven, and the people capacity are held on each vehicle. All the cars can carry a wheelchair in the boot. Some of the buses and minibuses are wheelchair accessible. Details of the vehicles are stored in *vehicles.txt*.

The company currently has six drivers. Each driver has an ID number, a name, address, phone number, the amount of mileage that they have done for the company to date, a start date and the type of license that they have. A B licence allows them to drive a taxi only, a D1 licence allows them to drive a minibus and a taxi, a D licence allows them to drive all three types of vehicles. Details are stored in *drivers.txt*.

When a customer contacts CallOut Dispatch Services to arrange a fare, they give their name, telephone number, source, destination and the number of people that will be travelling. They also indicate if any of the passengers use a wheelchair.

The dispatcher views the details on all vehicles to see if there is a suitable vehicle available for the fare. If so, he/she then views the details on all drivers to see if there is an available driver with a suitable licence. If a driver and vehicle are available, the dispatcher checks google maps to determine how long the journey will take (you will just enter this value). The dispatcher then schedules the fare consisting of the customer's details, the vehicle details and the journey details and these details are given to the chosen driver.

The dispatcher should be able to view a list of all active fares at any time. This should include details on the vehicle, driver and customer.

If a driver and/or vehicle are not available, the dispatcher tells the customer that there is no driver available at that time.

After each fare, a driver goes to the dispatcher and tells them that the fare is complete. The driver is marked as free and the fare details are added to the daily log. These details include the fare details that were in the driver's schedule and the cost of the fare (€5 call-out charge + € 6.50 per kilometre). The number of kilometres is updated for both the driver and the vehicle.

At the end of the day, a report is to be generated. For each vehicle and for each driver, it will show the number of kilometres driven and the revenue generated from fares.

You are required to design, code and test an Object Oriented application that will meet the requirements of Callout Dispatch Services. You are to implement a menu with (at least) the following options:

- Display all details on all vehicles (including whether the vehicle is free or not).
- Display all details on all drivers (including whether the driver is free or not).
- Display all active fares.

- Log return from fare.
- Allocate fare.
- Print daily report.
- Exit.

Furthermore, you are to use inheritance as effectively as possible and will have the final code as separate files .h and .cpp files.

SUBMIT TO MOODLE

1. **Full documentation of your algorithm:** This will include the preparatory work that you did before coding. It should contain details on the classes that you intend to implement and the members and methods that you intend including in each class. You are to include a diagram of the hierarchy of the classes that you are using and any inheritance that exists between them with any virtual inheritance marked clearly on the diagram. Specify the data type of each member and detail an algorithm for each of the methods and the main program. Include any other documentation that you consider relevant.

2. **Testing of your program**

   This should show how you tested your program. This should include at least four journeys that between them test all aspects of your program. In each case indicate the test that you are carrying out and the result of that test.

   Include 1. and 2. And the source code in one document, and upload that to **Skills Demo #2 – Documentation** on Moodle.

3. **The source code zipped.**

   Submit your source code (zipped) to **Skills Demo #2 – Source Code** on Moodle.

Marks will be allocated as shown on the following page.

Cork Education and Training Board

Object-Oriented Programing 6N2108	Learner Marking Sheet 2 Skills Demonstration #2 (Practical) 30%

Learner's Name:_____ Learner's PPSN:_____

Skills Demonstration 2		
Program design, to include at least: • Class design (with awareness of data abstraction and appropriate use of member and methods). • A hierarchy of the classes depicting the inheritance used to solve the devised problem. • Documentation of any issues that might result from the designed hierarchy and how such issues will be resolved. • Identification of any parts of the designed solution that might be reused. • Overall program design.	8	
Program implementation to include at least: • A parallel between design and implementation. • A final product with no syntax errors. • An attempt at each of the individual requirements of the devised brief.	12	
Quality of application to include at least: • A final product that meets all specifications in the brief. • A final product with no run-time errors. • A final product that adheres to industry standard best practices. • Robust data verification and validation with appropriate feedback. • Effective use of object oriented concepts to include inheritance. • Reusable parts of the programs saved as separate entities and linked to from within the source code.	6	
Testing of application to include at least: • A comprehensive set of test data (and expected results) that tests the program for correct input and incorrect input.	4	
Total Mark:	**30**	

Assessor's Signature:_____ Date:_____

External Authenticator's Signature:_____ Date:_____