# Statement of Completeness (Template): Mates Rates Rent-a-Car

This statement of completeness will need to *accurately* state the functionality which has been implemented. Use this checklist of program functionality as you complete your project.

**Student Name: Brendan Hutchins**

**Student ID: n10684085**

## Functionality Checklist

**In the following section, you are required to mark which functionality you have implemented. In the column on the right please mark ‘Y’ where you have completed this functionality, and ‘N’ where you have not. Please fill in any additional text boxes requested, and please note any limitations or bugs in the box at the end of each section. You may expand the table if you need more room for comments.**

|  |  |  |
| --- | --- | --- |
| **File I/O** | | |
| **File location** | *If your solution does not load files from the specified location in the specification, please note here where the files are stored.*  **Customers:** Data/customer.csv  **Fleet:** Data/fleet.csv  **Rentals:** Data/rentals.csv | |
| **Read** | CRM | Y/N |
| Fleet | Y/N |
| Rentals (must meet above criteria first) | Y/N |
| **Write** | CRM (must be able to read CRM first) | Y/N |
| Fleet (must be able to read CRM first) | Y/N |
| Rentals (must meet above criteria and read Rentals first) | Y/N |
| **Comments** | *Program catches any issues with files not existing or being open by another program* | |

*Please note that your assignment must be able to load the files in the same format as provided in the example files.*

|  |  |  |
| --- | --- | --- |
| **CRM And Fleet Functionality** | | |
| **CRM** | View customers | Y/N |
| Add customer   * With validation – all fields are valid options, no repeated customer ID | Y/N  Y/N |
| Remove customer   * With validation – only if not renting vehicle | Y/N  Y/N |
| Edit customer   * With validation – same as add | Y/N  Y/N |
| **Fleet** | View vehicles | Y/N |
| Add vehicle   * With validation – all fields are valid options, and there is no repeated registrations | Y/N  Y/N |
| Remove vehicle   * With validation – only if not already being rented | Y/N  Y/N |
| Edit vehicle   * With validation – same as add | Y/N  Y/N |
| *Extra marks – implemented vehicle classes with inheritance*  Please include a brief explanation of the inheritance here if implemented: Used an abstract base class for Vehicle and then an Enum for each VehicleGrade that has different defaults that refer to the Vehicle class to edit its values. | Y/N |
| **Renting** | View rented vehicles report. This should show the vehicle, customer, and the daily cost of the rental. | Y/N |
| Rent vehicle. This should show the total cost of the rental.   * With validation – only if vehicle and customer exist, vehicle is not already being rented, and customer is not already renting | Y/N  Y/N |
| Return vehicle.   * With validation – only if vehicle was being rented by the customer | Y/N  Y/N |
| **Comments** | *Rented report does not show daily cost of the rental, only registration and customer ID* | |
| **Changes from spec.** | *Please* ***briefly*** *note any changes from the specification of the classes and methods, and your reason for these.*  *I made the fields of Vehicle and Customer be public to better access those later when modifying those objects. I also had the Vehicle class be abstract to have the grades be extensions of the vehicle class. Finally, I added a method to return a vehicle of a specified registration ID called GetVehicle() in the Fleet Class.* | |

|  |  |  |
| --- | --- | --- |
| **Search Functionality** | | |
| **Possible to Query - Simple** | Any   * Be able to see any unrented vehicle in the fleet | Y/N |
| Single attribute query   * Example:   + Red | Y/N |
| A choice between two attributes (disjunction)   * Example:   + Family OR Luxury | Y/N |
| A combination of two attributes (conjunction)   * Example:   + Family AND Luxury | Y/N |
| **Inter-mediate** | A choice of any number of attributes (disjunction)   * Example:   + Family OR Luxury OR Red | Y/N |
| A combination of any number of attributes (conjunction)   * E.g.:   + Luxury AND Red | Y/N |
| **Advanced** | A combination of any number of attributes, using both AND and OR, where the operators AND and OR should have the same priority (precedence)   * E.g.:   + Economy OR Family AND 4-Cylinders | Y/N |
| ***Ninja*** | A combination of any number of attributes, using AND and OR, with AND having a higher priority than OR, supporting parenthesis to resolve priority   * E.g.:   + ((GPS AND Sunroof) OR (Red OR Green)) AND Commercial OR Luxury | Y/N |
| **Comments** | *Sometimes Shunting yard/parse text does not catch extra parenthesis, but later SearchVehicles does, so sometimes results in incorrect error mess* | |

## Program Transcript

***You are also required to attach a full transcript demonstrating all of the features of your program (you can use the checklist above as a basis). You will practice this in a future workshop. You must include this transcript as part of your submission. Please state the name of the transcript file.***

**Transcript File Name:**

***transcript.txt***