|  |
| --- |
|  |
| Fourth Assignment |
| Inheritance, interfaces and exceptions |

|  |
| --- |
| Kevin de la Coba Malam  Marcos Bernuy López |

**Section 1**

In this section we first created the folder vehicle which will be the package where the interface IVehicle with the classes which will implemented where situated. These classes which we created represent the vehicles which must be created after reading the txt file. We decided to create an abstract class called Vehicle which implements the interface and from which it inherits into de subclasses Motorcycle, Truck and Car. We decided to create an abstract class, to decrease the repetition of code, since the vehicles implemented part of the functions exactly the same way. It was also needed to create the id of the vehicles created which needed a private static variable shared by all vehicles. Once this classes were implemented, we implemented the class RaceReader which implements the function read() which reads the data from the txt file and returns a Race, an object from the Race class. We implement this function making it read each line splitting the words received, this way we can get the quantity, and the maximum speed of each vehicle. Then we just make a loop to create each of the vehicles adding them to an ArrayList. Once we have the ArrayList and the length of the race, we can create the race which will be the return of the function read() Finally in this part we implement the exceptions package in which we implement the RaceException.java which extends from the Exception class so not many changes are needed.

**Section 2**

In this part we have to implement the function simulate() from the Race class, which simulates the race. In this simulate function we had to take a couple things into account. First of all, we had to implement the probability of the vehicles which participate in the race as it is asked. To make this possible, we implemented the function getRealSpeed() in the Vehicle class and its subclasses which returns a double with the speed after using the Math.random() function to represent the probability of getting the maximum speed. Once the real speed was calculated, we just had to change the actual position to its position plus the actual speed. Afterwards we made sure the output was printed correctly and exactly as the example given.

**Section 3**

**Section 4**

**Section 5**

**Class Diagram**