# Maintenance

## Classes

I have written my program in 9 classes and an enum. These are each described below.

|  |  |
| --- | --- |
| **Class Name** | **Purpose** |
| CarDealership | The class that stores the main method. This also creates the original frame that is used to hold the GUI for the entire app.  CarDealership implements ActionLister so that the actionPerformed method can be overwritten to hold all of the user interactions for the entire app and manage the different JPanels linking together. |
| Car | A blueprint for the car object, the cars will be stored in an ArrayList of these objects  Car extends SearchableObject so it can be searched. |
| Customer | A blueprint for the customer object, the customers will be stored in an ArrayList of Customers.  Customer extends SearchableObject so it can be searched. |
| Employee | A blueprint for the employee object, employees are stored in an ArrayList of Employees.  Employee extends SearchableObject so it can be searched |
| SearchableObject | A basic object that the Car, Customer and Employee classes are based off of. This only has two attributes, one for name (which is the field that will be searched) and one for type; this is where the type of the extended object is stored. |
| Type (enum) | This is the type field that I mentioned above, this enumeration only contains 3 fields: CAR, EMPLOYEE and CUSTOMER |
| Login | This is a GUI class that was designed in NetBeans, this will contain a JPanel that holds the components required to login.  Login extends JPanel so that is can be easily added to the main JFrame in the CarDealership class |

## Methods

Below is a table showing the methods I have used for each class.

|  |  |  |
| --- | --- | --- |
| Class | Method | What this method does |
| Main | playAgain() | This method will ask the user if they would like to play the game again. If so it return a Boolean |
| Main | Main() | This is the main class that will create and initialise all of the objects used in the program. |
| numberGenerator | numberGenerator() | This is the constructor for this class, it simply initialises the random object to allow the generation of a number |
| numberGenerator | generate() | This method will generate a random number and store it in the generatedNumber variable. |
| inputANumber | getNumber() | This method will ask the user to enter a number; it handles any errors cause by entering non number characters. This method will return an integer of the number that has been entered. |
| numberChecker | isNumberCorrect() | This method handles the checking of numbers to see if they match the generated number. This method will return a Boolean value if the number is correct. |
| numberChecker | higherOrLower() | This method is called by the above method if the number is incorrect is simply outputs if the number is too high or too low. |

## Imports

I have used multiple imports within this project these are listed below

import java.util.Scanner;

This imports the Scanner java utility which will allow me to create a Scanner object to read user input

import static java.lang.System.in;

This is a static import of System.in, this allows me to reduce my code as I no longer have to type System when using the *in* object.

import static java.lang.System.out;

This is a static import of System.out, this allows me to reduce my code as I no longer have to type System when using the *out* object.

import java.util.InputMismatchException;

This is an import of the InputMismatchException, this is used within the inputANumber class and allows me to use a try and catch to ensure that the input that the user is giving matches an integer.

import java.util.Random;

Finally I have also imported the Random java utility to allow the creation of the random object and the generation of random numbers.

## Hardware and software used

For the development of my program I used IntelliJ IDEA, this is an IDE that allows you to write Java programs is an effect environment and compile and execute them in real time. When maintaining this program you should try and use the same IDE as it was originally developed with. This means that any snippets of code should still work in the same way and you shouldn’t see differences in performance or the functionality of different methods.

IntelliJ IDEA can be downloaded from the official jet brains website [www.jetbrains.com/idea/](http://www.jetbrains.com/idea/)

The minimum requirements to open this program in IntelliJ are as follows.

* 8GB RAM
* Windows 64-bit Operating System
* JDK 7