**FOP 2 - Lab 9**

1. Type and compile the examples from the lecture.
2. Modify the constructor and the set methods from the Car class (created in the previous lab) so that the make and registration cannot take the values of empty strings, while the engine size should take a value within 0.1 and 7.0 range; if empty strings and double values outside that range are passed, the instance variables should take the values of “Dacia”, “191-D-1”, and 0.1, respectively. Ensure that you also modify the main method to test for all these cases.
3. Modify the class CarShow to do the following:

* Ask user how many cars are in the show (at least 5);
* Then create as many cars as the user stated in previous step and add them to the list.
* When asked to display the contents of the list, use both an enhanced for loop and a traditional for loop.

1. Write a small program that contains the following static methods (ensure that you test these methods with various strings by calling them in the main method); for this exercise you can assume that the strings/sentences contain only words delimited by a space, and it contains no punctuation, so you can tokenise the sentence using the space/white spaces only:

* Tokenises a string/sentence and returns the number of occurrences of the word *car,* irrespective of its case; for example: if the sentence “Cars are to be stored in a car yard with lots of other CAR related things” is passed, the method returns 2.
* Tokenises a string/sentence and prints only the words that end with *ly;* for instance, if the sentence “SERIOUSLY, you certainly need to study vigorously for your practical exam” is passed, the method prints *certainly vigorously*
* Takes in a string/sentence, replaces all the instances of a space with “\_”, and then prints that string; for example, if the string “Hello there, are you prepared for exam?” should print Hello\_\_\_there­,\_are\_you\_prepared\_for\_\_exam?
* Takes in a string/sentence, finds out all the indices of characters “e”, places them into a list and returns that list; for example, if the string “Well, there are three cars in the show” is passed, the method returns the list [1, 8, 10, 14, 19, 20, 32].
* Takes in 2 strings/sentences and prints which one comes first alphabetically; if the strings are the same print “either one can be first”; for example, if *string* and *strong* are passed, the method prints “string comes before strong”; if *string* and *string* are passed, the method prints “either one can be first”
* Takes in a list of sentences and prints them backwards; for example, if the list [“FOP2 and Java”,”Fop1 and Python”] is passed, the method prints:

*avaJ dna 2POF*

*nohtyP dna 1poF*