Exercise 1

Create public class ***ParkingSpace*** with the following elements:

- Private attribute *free* that indicates whether the parking space is free; it has the value *true* if the parking space is free, otherwise its value is *false*.
- Private attribute *registrationNumber* that represents the registration number of the vehicle parked at that space (type String).
- Set and get methods for these two attributes.

Create public class ***ParkingLot*** with the following elements:

- Private attribute *parkingSpaces,* an array of objects of the *ParkingSpace* class.
- Public constructor having the capacity of the parking lot (i.e., the total number of parking spaces) as its input parameter. If the value of this parameter is greater than zero, the *parkingSpaces* array should be initialized to that value. Otherwise (i.e., if the value of the input parameter is less than or equal to zero), the capacity of the parking lot should be set to 40 and an error message should be printed to the screen. In either case, all parking spaces (i.e., the elements of the *parkingSpaces* array) should be initialized and set to be free.
- Public method which checks if there is a free parking space; the method returns true if there is at least one free parking space, otherwise it returns false.
- Public method for 'parking' a vehicle. The input parameter of this method is the registration number of the vehicle to be parked. One should first check if there are still free spaces at the parking lot; if yes, the vehicle should be 'parked' at the first free space; if not, an appropriate message should be printed on the screen.
- Public method which checks if the vehicle with the given registration number is at the parking lot; the method returns true is the vehicle is found, otherwise it returns false; the registration number of the vehicle to be checked is passed to the method as its input parameter. *(students work on their own)*
- Public method for removing a vehicle from the parking lot. The input parameter of the method is the registration number of the vehicle to be removed. One should first check if there is a vehicle with the given registration number at the parking lot; if yes, the vehicle should be removed, so that the parking space becomes free again; if not, an appropriate message should be printed on the screen. Removing a vehicle assumes that the parking space occupied by that vehicle is set free and the data about vehicle's registration number are destroyed. *(students work on their own)*

Create the ***TestParkingLot*** class. In the main method of this class create an object of the class *ParkingLot* with the capacity of 20 parking spaces; park two vehicles with the following registration numbers: "BG 123-456" and "NS 234-56".

Exercise 2

Create public class ***Motorcycle*** with the following elements:

- Private (String) attribute *brand*
- Private (String) attribute *model*
- Private (int) attribute *enginePower*.
- Get and set methods for these three attributes.

This class should also:

- Redefine the method *toString* inherited from the Object class; the method should return a string containing values of all the attributes.
- Redefine the *equals* method inherited from the Object class. This method returns true if values of all three attributes (brand, model, enginePower) of the two objects that are compared are equal; otherwise it returns false.

Create public class ***MotorcycleRegistry*** with the following elements:

- Private attribute *motorcycles* that represents a list of objects of the *Motorcycle* class
- Public constructor that initializes the *motorcycles* attribute.
- Public method *isInRegistry* that receives an object of the class Motorcycle (as its input parameter) and checks if the same motorcycle is already in the registry.
- Public method *addToRegistry* that receives an object of the class Motorcycle (as its input parameter); the method should first check that the input parameter is not null and the registry does not already contain the same motorcycle; if both requirements are fulfilled the new motorcycle is added to the registry. *(students work on their own)*
- Public method *deleteFromRegistry* which receives an object of the class Motorcycle (as its input parameter) and deletes it from the registry; the deletion should take place only if the motorcycle to be deleted is in the registry. *(students work on their own)*
- Public method *print* which prints data about all motorcycles in the registry.

Create class ***TestMotorcyclesRegistry***. In the main method of this class create an object of the *MotorcycleRegistry* class and three objects of the *Motorcycle* class using the following values for their attributes: “Honda” - “CB 750 F” - 748, “Kawasaki” - “ER 5” - 498 i “Honda” - “CB 750 F” - 748. First add the first two objects to the registry and then print the content of the registry. Subsequently check if the third motorcycle is in the registry and if yes, delete it from the registry. Finally, print again the content of the registry.