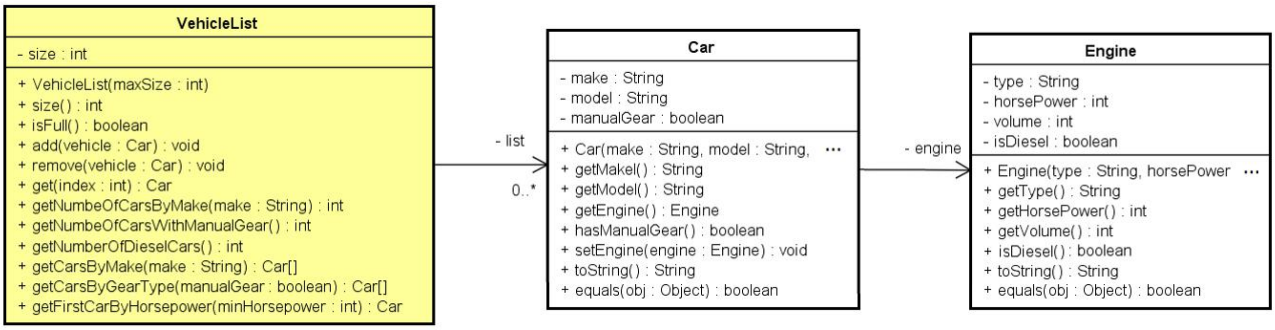
Exercise: VehicleList (a list of Car's), version 1

Study the class diagram below (note that constructors in class `Car` and `Engine` are not fully shown)


VehicleList
- size : int
+ VehicleList(maxSize : int)
+ size() : int
+ isFull() : boolean
+ add(vehicle : Car) : void
+ remove(vehicle : Car) : void
+ get(index : int) : Car
+ getNumbeOfCarsByMake(make : String) : int
+ getNumbeOfCarsWithManualGear() : int
+ getNumberOfDieselCars() : int
+ getCarsByMake(make : String) : Car[]
+ getCarsByGearType(manualGear : boolean) : Car[]
+ getFirstCarByHorsepower(minHorsepower : int) : Car
- list
0..*
Car
- make : String
- model : String
- manualGear : boolean
+ Car(make : String, model : String, ...
+ getMakel() : String
+ getModel() : String
+ getEngine() : Engine
+ hasManualGear() : boolean
+ setEngine(engine : Engine) : void
+ toString() : String
+ equals(obj : Object) : boolean
- engine
Engine
- type : String
- horsePower : int
- volume : int
- isDiesel : boolean
+ Engine(type : String, horsePower ...
+ getType() : String
+ getHorsePower() : int
+ getVolume() : int
+ isDiesel() : boolean
+ toString() : String
+ equals(obj : Object) : boolean

Create a module named `VehicleList_v1` and copy classes `Car` and `Engine` into this module.

Either copy the classes `Car` and `Engine` given at the end of this document, or reuse your own version of the two classes implemented with association. You do not need the exact same instance variables and methods but make sure you have method `equals` in both classes, `isDiesel` and `getHorsePower` in class `Engine`, and methods `getMake`, `hasManualGear` and `getEngine` in class `Car`.

Create a class `VehicleList` with:

a) A `Car` array instance variable, i.e. of type `Car[]`.

b) An `int` instance variable for the actual size (number of objects added to the array).

c) A constructor initializing the array to a max size (given as an argument) and initialising the `size` instance variable to 0.

d) A method `size` returning the actual size of the list (not the capacity).

e) A method `isFull` returning true is the list/array is full.

f) A method `add`, adding a car to end of the list. If the list is full, nothing is added.

g) A method `remove`, removing the element at the index specified – by shifting the following elements one back. If the index is out of bounds, no elements are removed.

h) A method `get` returning the element at the index specified.

i) A method `getNumberOfCarsByMake`, returning the number of cars with a make equal to the specified parameter variable.

j) A method `getNumberOfCarsWithMAnualGear`, returning the number of cars with a manual gear.

k) A method `getNumberOfDieselCars`, returning the number of cars with a diesel engine.

l) A method `getCarsByMake`, returning an array with each car having the make equal to the parameter variable specified. The size of the array is equal to the returned value from the method `getNumberOfCarsByMake`.

m) A method `getCarsByGearType`, returning an array with each car having either manual gear or not depending on the value of the parameter variable. The size of the array can be found with the use of the method `getNumberOfCarsWithMAnualGear`.

n) A method `getFirstCarByHorsepower`, returning a reference to a `Car` object an engine's horsepower at least the value specified with the parameter variable `minHorsepower`. If no cars are found, then the method returns `null`.

Create a test class with a `main` method and test the class `VehicleList` thoroughly.

Class Engine

```
public class Engine
{
  private String type;
  private int horsePower;
  private int volume;
  private boolean isDiesel;

  public Engine(String type, int horsePower, int volume, boolean isDiesel)
  {
    if (type == null)
    {
      type = "";
    }
    this.type = type;
    this.horsePower = horsePower;
    this.volume = volume;
    this.isDiesel = isDiesel;
  }

  public String getType()
  {
    return type;
  }

  public int getHorsePower()
  {
    return horsePower;
  }

  public int getVolume()
  {
    return volume;
  }

  public boolean isDiesel()
  {
    return isDiesel;
  }

  @Override public String toString()
  {
    String s = type + ", " + volume + " cm3, " + horsePower + " hp, ";
    if (!isDiesel)
    {
      s += "not a";
    }
    s += " diesel";
    return s;
  }

  @Override public boolean equals(Object obj)
  {
    if (!(obj instanceof Engine))
    {
      return false;
    }
    Engine other = (Engine) obj;
    return horsePower == other.horsePower && volume == other.volume
        && isDiesel == other.isDiesel && type.equals(other.type);
  }
}
```

Class Car

```
public class Car
{
  private String make;
  private String model;
  private boolean manualGear;
  private Engine engine;

  public Car(String make, String model, Engine engine, boolean manualGear)
  {
    if (make == null)
    {
      make = "NO make";
    }
    if (model == null)
    {
      model = "No model";
    }
    this.make = make;
    this.model = model;
    this.manualGear = manualGear;
    setEngine(engine);
  }

  public String getMake()
  {
    return make;
  }

  public String getModel()
  {
    return model;
  }

  public boolean hasManualGear()
  {
    return manualGear;
  }

  public Engine getEngine()
  {
    return engine;
  }

  public void setEngine(Engine engine)
  {
    this.engine = engine;
  }

  @Override public String toString()
  {
    String s = make + " " + model + ", ";
    if (manualGear) {s += " manual gear";   }
    else {s += "automatic gear";}

    s += ", " + engine;
    return s;
  }

  @Override public boolean equals(Object obj)
  {
    if (!(obj instanceof Car))
    {
      return false;
    }
    Car other = (Car) obj;
    return make.equals(other.make) && model.equals(other.model) && engine
        .equals(other.engine) && manualGear == other.manualGear;
  }
}
```