OO Challenge

Part One

Create a class for vehicle. Each vehicle instance should have the following properties:

- *make*
- *model*
- *year*

Each vehicle instance should have access to a method called ***honk***, which returns the string "Beep."

```
let myFirstVehicle = new Vehicle("Honda", "Monster Truck", 1999);
myFirstVehicle.honk(); // "Beep."
```

Each vehicle instance should have a method called toString, which returns the string containing the make, model and year.

```
let myFirstVehicle = new Vehicle("Honda", "Monster Truck", 1999);
myFirstVehicle.toString(); // "The vehicle is a Honda Monster Truck
from 1999."
```

```
class Vehicle {
  constructor(make, model, year) {
    this.make = make;
    this.model = model;
    this.year = year;
  }
  honk() {
    return "Beep!";
  }
  toString() {
    return `The vehicle is a ${this.make} ${this.model} from ${thi
s.year}`;
  }
}
```

Part Two

Create a class for a car. The ***Car*** class should inherit from ***Vehicle*** and each car instance should have a property called ***numWheels*** which has a value of 4.

```
let myFirstCar = new Car("Toyota", "Corolla", 2005);
myFirstCar.toString(); // "The vehicle is a Toyota Corolla from 2005."
myFirstCar.honk();     // "Beep."
myFirstCar.numWheels;  // 4
```

```
class Car extends Vehicle {
  constructor(make, model, year) {
    super(make, model, year);
    this.numWheels = 4;
  }
}
```

Part Three

Create a class for a Motorcycle. This class should inherit from ***Vehicle*** and each motorcycle instance should have a property called ***numWheels*** which has a value of 2. It should also have a ***revEngine*** method which returns "VROOM!!!"

```
let myFirstMotorcycle = new Motorcycle("Honda", "Nighthawk", 2000);

myFirstMotorcycle.toString();
// "The vehicle is a Honda Nighthawk from 2000."

myFirstMotorcycle.honk();     // "Beep."
myFirstMotorcycle.revEngine(); // "VROOM!!!"
myFirstMotorcycle.numWheels;  // 2
```

```
class Motorcycle extends Vehicle {
  constructor(make, model, year) {
    super(make, model, year);
    this.numWheels = 2;
  }
  revEngine() {
    return "VROOM";
  }
}
```

Part Four

Create a class for a Garage. It should have a property called ***vehicles*** which will store an array of vehicles, and a property called ***capacity*** which is a number indicating how many vehicles will fit in the garage. When you create a garage, ***vehicles*** will always be empty; you only need to provide the ***capacity***.

A garage should also have an ***add*** method, which attempts to add a vehicle to the array of vehicles. However, if you try to add something which is *not* a vehicle, the garage should return the message “Only vehicles are allowed in here!”. Also, if the garage is at capacity, it should say “Sorry, we’re full.”

```
let garage = new Garage(2);
garage.vehicles; // []
garage.add(new Car("Hyundai", "Elantra", 2015)); // "Vehicle added!"
garage.vehicles; // [Car]
garage.add("Taco"); // "Only vehicles are allowed in here!"

garage.add(new Motorcycle("Honda", "Nighthawk", 2000));
// "Vehicle added!"
garage.vehicles; // [Car, Motorcycle]

garage.add(new Motorcycle("Honda", "Nighthawk", 2001));
// "Sorry, we're full."
```

```
class Garage {
  constructor(capacity) {
    if (!Number.isFinite(capacity)) {
      throw new Error("Capacity must be numeric");
    }
    this.capacity = capacity;
    this.vehicles = [];
  }
  add(vehicle) {
    if (!(vehicle instanceof Vehicle)) {
      return "Only vehicles are allowed here!";
    } else if (this.vehicles.length >= this.capacity) {
      return "Sorry we're full.";
    } else {
      this.vehicles.push(vehicle);
      return "Vehicle added!";
    }
  }
}
```

Solution

See Our solution.