Develop simple java and JS based program to show is-a, has-a, uses-a relationship

```
class Vehicle {
    String type;
    Vehicle(String type) {
        this.type = type;
    }
    void displayInfo() {
        System.out.println("Type: " + this.type);
    }
}

class Car extends Vehicle {
    String brand;
    Car(String type, String brand) {
        super(type);
        this.brand = brand;
    }
    void displayInfo() {
        super.displayInfo();
        System.out.println("Brand: " + this.brand);
    }
}

class Garage {
    Vehicle[] vehicles;
    Garage(Vehicle[] vehicles) {
        this.vehicles = vehicles;
    }
    void displayAllVehicles() {
        System.out.println("All Vehicles in Garage:");
        for (Vehicle vehicle : vehicles) {
```

```
            vehicle.displayInfo();
        }
    }
}

class Mechanic {
    Garage garage;
    Mechanic(Garage garage) {
        this.garage = garage;
    }
    void repairVehicle(String type) {
        System.out.println("Repairing vehicle of type: " + type);
        for (Vehicle vehicle : garage.vehicles) {
            if (vehicle.type.equals(type)) {
                System.out.println("Vehicle of type " + type + " repaired.");
                return;
            }
        }
        System.out.println("No vehicle of type " + type + " found in the garage.");
    }
}

public class Main {
    public static void main(String[] args) {
        Vehicle car1 = new Car("Car", "Toyota");
        Vehicle car2 = new Car("Car", "Honda");
        Vehicle bike1 = new Vehicle("Bike");
        Vehicle[] vehicles = {car1, car2, bike1};
        Garage garage = new Garage(vehicles);
        Mechanic mechanic = new Mechanic(garage);
        garage.displayAllVehicles();
```

```
        mechanic.repairVehicle("Car");
        mechanic.repairVehicle("Truck");
    }
}
```

Java script

```
class Vehicle {
    constructor(type) {
        this.type = type;
    }
    displayInfo() {
        console.log("Type: " + this.type);
    }
}

class Car extends Vehicle {
    constructor(type, brand) {
        super(type);
        this.brand = brand;
    }
    displayInfo() {
        super.displayInfo();
        console.log("Brand: " + this.brand);
    }
}

class Garage {
    constructor(vehicles) {
        this.vehicles = vehicles;
    }
    displayAllVehicles() {
```

```
        console.log("All Vehicles in Garage:");
        this.vehicles.forEach(vehicle => {
            vehicle.displayInfo();
        });
    }
}

class Mechanic {
    constructor(garage) {
        this.garage = garage;
    }
    repairVehicle(type) {
        console.log("Repairing vehicle of type: " + type);
        for (let vehicle of this.garage.vehicles) {
            if (vehicle.type === type) {
                console.log("Vehicle of type " + type + " repaired.");
                return;
            }
        }
        console.log("No vehicle of type " + type + " found in the garage.");
    }
}

const main = () => {
    const car1 = new Car("Car", "Toyota");
    const car2 = new Car("Car", "Honda");
    const bike1 = new Vehicle("Bike");
    const vehicles = [car1, car2, bike1];
    const garage = new Garage(vehicles);
    const mechanic = new Mechanic(garage);
    garage.displayAllVehicles();
```

```
    mechanic.repairVehicle("Car");
    mechanic.repairVehicle("Truck");
};
main();
```