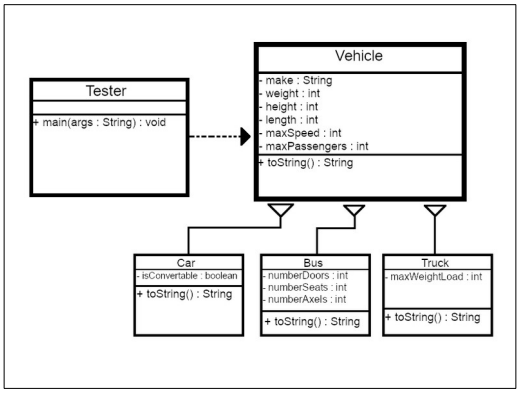
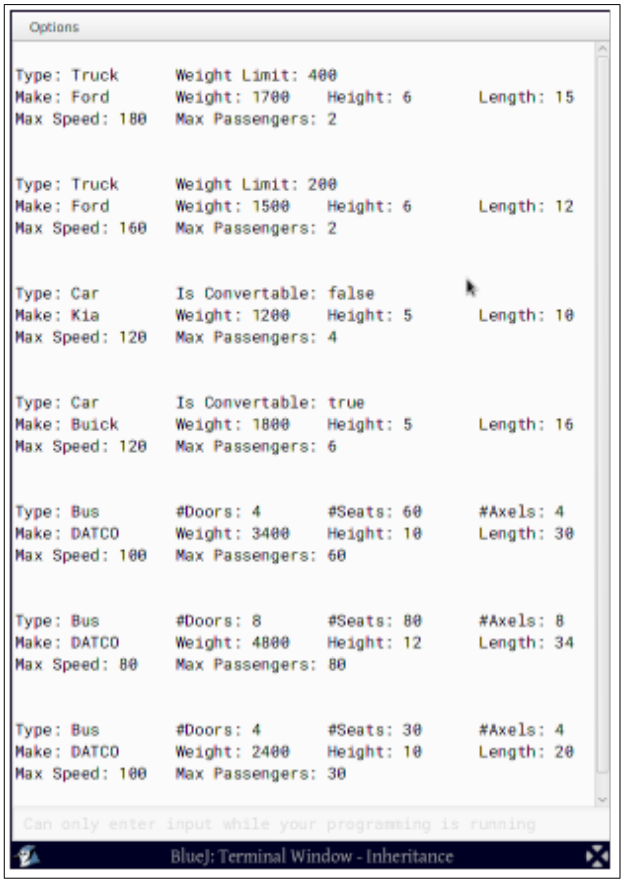
UML



Output



```
Options

Type: Truck      Weight Limit: 400
Make: Ford       Weight: 1700    Height: 6       Length: 15
Max Speed: 180   Max Passengers: 2


Type: Truck      Weight Limit: 200
Make: Ford       Weight: 1500    Height: 6       Length: 12
Max Speed: 160   Max Passengers: 2


Type: Car        Is Convertable: false
Make: Kia        Weight: 1200    Height: 5       Length: 10
Max Speed: 120   Max Passengers: 4


Type: Car        Is Convertable: true
Make: Buick      Weight: 1800    Height: 5       Length: 16
Max Speed: 120   Max Passengers: 6


Type: Bus        #Doors: 4       #Seats: 60      #Axels: 4
Make: DATCO      Weight: 3400    Height: 10      Length: 30
Max Speed: 100   Max Passengers: 60


Type: Bus        #Doors: 8       #Seats: 80      #Axels: 8
Make: DATCO      Weight: 4800    Height: 12      Length: 34
Max Speed: 80    Max Passengers: 80


Type: Bus        #Doors: 4       #Seats: 30      #Axels: 4
Make: DATCO      Weight: 2400    Height: 10      Length: 20
Max Speed: 100   Max Passengers: 30

Can only enter input while your programming is running
BlueJ: Terminal Window - Inheritance
```

Tester.java

```
/**
 * @author (Kyle Guarco)
 * @version (July 13, 2020)
 */
public class Tester
{
    public static void main(String[] args)
    {
        // Vehicle constructor parameters are as follows (For reference; Switching tabs is so
exhausting):
        // String make, int weight, int height, int length, int maxSpeed, int maxPassengers
        Vehicle v;

        v = new Truck(400, "Ford", 1700, 6, 15, 180, 2);
        print(v);

        v = new Truck(200, "Ford", 1500, 6, 12, 160, 2);
        print(v);

        v = new Car(false, "Kia", 1200, 5, 10, 120, 4);
        print(v);

        v = new Car(true, "Buick", 1800, 5, 16, 120, 6);
        print(v);

        v = new Bus(4, 60, 4, "DATCO", 3400, 10, 30, 100);
        print(v);

        v = new Bus(8, 80, 8, "DATCO", 4800, 12, 34, 80);
        print(v);

        v = new Bus(4, 30, 4, "DATCO", 2400, 10, 20, 100);
        print(v);
    }

    // Function "alias". I didn't feel like typing 'System.out.println' 7 times.
    public static void print(Object s)
    {
        System.out.println("\n" + s);
    }
}
```

Vehicle.java

```
/**
 * This class represents the basic idea of anything that can
 * transport people on land.
 *
 * @author (Kyle Guarco)
 * @version (July 13, 2020)
 */
public abstract class Vehicle
{
    private String make;
    private int weight, height, length;
    private int maxSpeed, maxPassengers;

    public Vehicle(String make, int weight, int height, int length, int maxSpeed, int
maxPassengers)
    {
        this.make = make;
        this.weight = weight;
        this.height = height;
        this.length = length;
        this.maxSpeed = maxSpeed;
        this.maxPassengers = maxPassengers;
    }

    @Override
    public String toString()
    {
        return String.format("Make: %s\t Weight: %d\t Height: %d\t Length: %d\n" +
                             "Max Speed: %d\t Max Passengers: %d\n",
                             make, weight, height, length, maxSpeed, maxPassengers);
    }
}
```

Car.java

```
/**
 * @author (Kyle Guarco)
 * @version (July 13, 2020)
 */
public class Car extends Vehicle
{
    private boolean isConvertable;

    public Car(boolean isConvertable,
               String make, int weight, int height, int length, int maxSpeed, int maxPassengers)
    {
        super(make, weight, height, length, maxSpeed, maxPassengers);

        this.isConvertable = isConvertable;
    }

    @Override
    public String toString()
    {
        return String.format("Type: Car\t Is Convertable: %b\n", isConvertable)
                        // Call Vehicle.toString() and append it to the final string here.
                        + super.toString();
    }
}
```

Truck.java

```
/**
 * @author (Kyle Guarco)
 * @version (July 13, 2020)
 */
public class Truck extends Vehicle
{
    private int maxWeightLoad;

    public Truck(int maxWeightLoad,
                 String make, int weight, int height, int length, int maxSpeed, int maxPassengers)
    {
        super(make, weight, height, length, maxSpeed, maxPassengers);

        this.maxWeightLoad = maxWeightLoad;
    }

    @Override
    public String toString()
    {
        return String.format("Type: Truck\t Weight Limit: %d\n", maxWeightLoad)
                        // Call Vehicle.toString() and append it to the final string here.
                        + super.toString();
    }
}
```

Bus.java

```
/**
 * @author (Kyle Guarco)
 * @version (July 13, 2020)
 */
public class Bus extends Vehicle
{
    private int numberDoors, numberSeats, numberAxels;

    public Bus(int numberDoors, int numberSeats, int numberAxels,
               String make, int weight, int height, int length, int maxSpeed)
    {
        // On a bus, the maximum amount of passengers is equal to the number of seats.
        // So for 'Bus', we'll pass the number of seats into the parent constructor.
        super(make, weight, height, length, maxSpeed, numberSeats);

        this.numberDoors = numberDoors;
        this.numberSeats = numberSeats;
        this.numberAxels = numberAxels;
    }

        @Override
    public String toString()
    {
        return String.format("Type: Bus\t #Doors: %d\t #Seats: %d\t #Axels: %d\n",
                             numberDoors, numberSeats, numberAxels)
                             // Call Vehicle.toString() and append it to the final string here.
                             + super.toString();
    }
}
```