

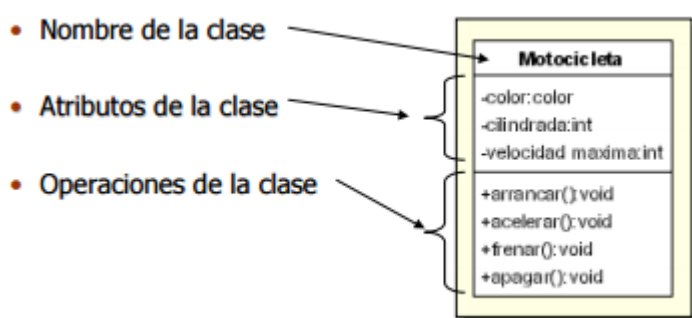
CARRERA	CURSO	AMBIENTE
Ingeniería de Sistemas e Informática	PROGRAMACIÓN ORIENTADA A OBJETOS	

PRACTICA No	NOMBRE DE PRACTICA	CODIGO DE LAB.	DURACION (HORAS)
01	Clases y Objetos	C206	2
Elaborado por Docente	Revisado por Jefe de Lab.	Aprobado por Coordinador	Autorizado por Director
Duilio Chavez Cuarite			

REVISION	FECHA	DESCRIPCION
1		Clases y Objetos

## IMPLEMENTANDO CLASES Y OBJETOS EN UN LENGUAJE OO

### Partes de una clase



### EJEMPLO

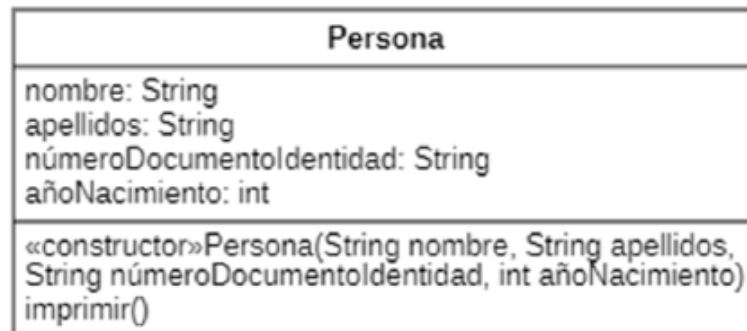
#### 1. ENUNCIADO: Clase Persona

Se requiere un programa que modele el concepto de una persona. Una persona posee nombre, apellido, número de documento de identidad y año de nacimiento. La clase debe tener un constructor que inicialice los valores de sus respectivos atributos. La clase debe incluir los siguientes métodos:

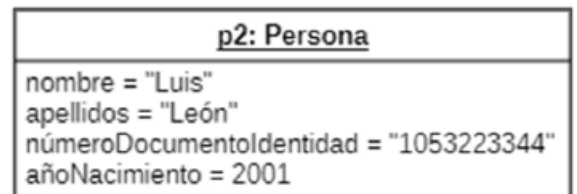
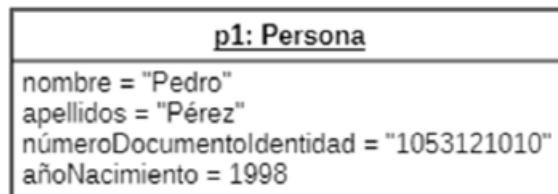
- Definir un método que imprima en pantalla los valores de los atributos del objeto.
- En un método main se deben crear dos personas y mostrar los valores de sus atributos en pantalla.

### Solución:

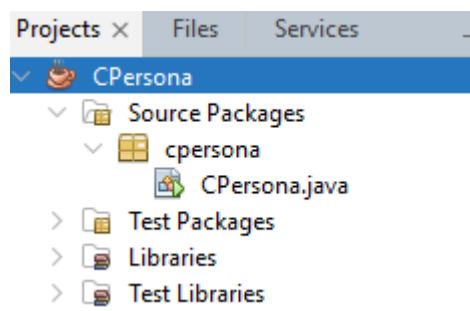
- Diagrama de clases



- Diagrama de objetos



- Proyecto



- Código fuente:

```

1 package cpersona;
2
3 public class CPersona {
4
5     String nombre;
6     String apellidos;
7     String numeroDocumentoIdentidad;
8     int añoNacimiento;
9

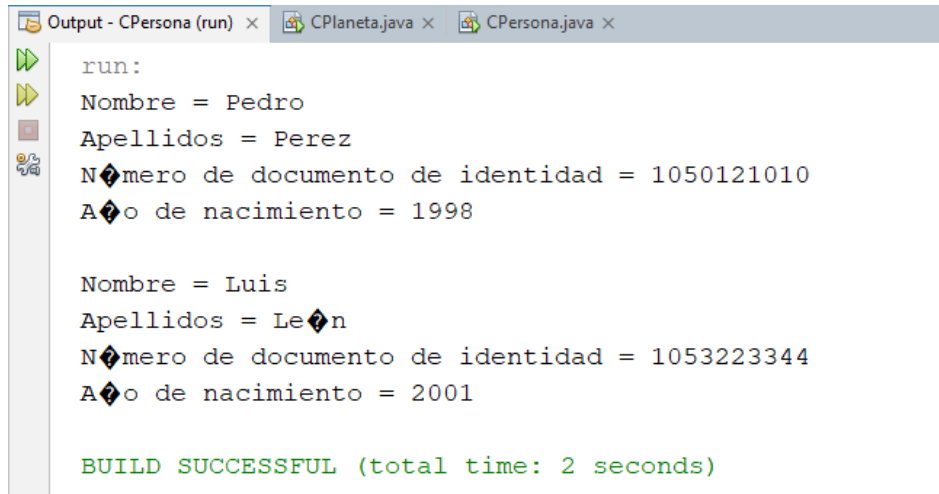
```

```

10 //Constructor
11 CPersona(String nombre, String apellidos, String numeroDocumentoIdentidad, int añoNacimiento){
12     this.nombre = nombre;
13     this.apellidos = apellidos;
14     this.numeroDocumentoIdentidad = numeroDocumentoIdentidad;
15     this.añoNacimiento = añoNacimiento;
16 }
17 // Método que imprime en pantalla los datos de una persona
18 void imprimir(){
19     System.out.println("Nombre = " + nombre);
20     System.out.println("Apellidos = " + apellidos);
21     System.out.println("Número de documento de identidad = " + numeroDocumentoIdentidad);
22     System.out.println("Año de nacimiento = " + añoNacimiento);
23     System.out.println();
24 }
25 public static void main(String[] args) {
26     CPersona p1 = new CPersona("Pedro", "Perez", "1050121010", 1998);
27     CPersona p2 = new CPersona("Luis", "León", "1053223344", 2001);
28     p1.imprimir();
29     p2.imprimir();
30 }
31 }
32

```

## Resultado:



```

Output - CPersona (run) x CPlaneta.java x CPersona.java x
run:
Nombre = Pedro
Apellidos = Perez
Número de documento de identidad = 1050121010
Año de nacimiento = 1998

Nombre = Luis
Apellidos = León
Número de documento de identidad = 1053223344
Año de nacimiento = 2001

BUILD SUCCESSFUL (total time: 2 seconds)

```

## 2. ENUNCIADO: Clase Planeta

Se requiere un programa que modele el concepto de un planeta del sistema solar. Un planeta tiene los siguientes atributos:

- Un nombre de tipo String con valor inicial de null.
- Cantidad de satélites de tipo int con valor inicial de cero.
- Masa en kilogramos de tipo double con valor inicial de cero.
- Volumen en kilómetros cúbicos de tipo double con valor inicial de cero
- Diámetro en kilómetros de tipo int con valor inicial de cero.

- Distancia media al Sol en millones de kilómetros, de tipo int con valor inicial de cero.
- Tipo de planeta de acuerdo con su tamaño, de tipo enumerado con los siguientes valores posibles: GASEOSO, TERRESTRE y ENANO.
- Observable a simple vista, de tipo booleano con valor inicial false.

**La clase debe incluir los siguientes métodos:**

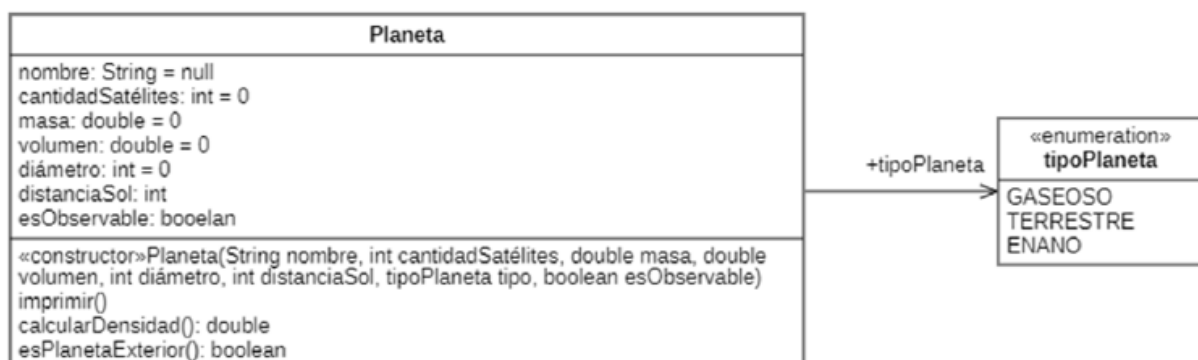
- La clase debe tener un constructor que inicialice los valores de sus respectivos atributos.
- Definir un método que imprima en pantalla los valores de los atributos de un planeta.
- Calcular la densidad un planeta, como el cociente entre su masa y su volumen.
- Determinar si un planeta del sistema solar se considera exterior.

Un planeta exterior está situado más allá del cinturón de asteroides. El cinturón de asteroides se encuentra entre 2.1 y 3.4 UA. Una unidad astronómica (UA) es la distancia entre la Tierra y el Sol= 149 597 870 Km.

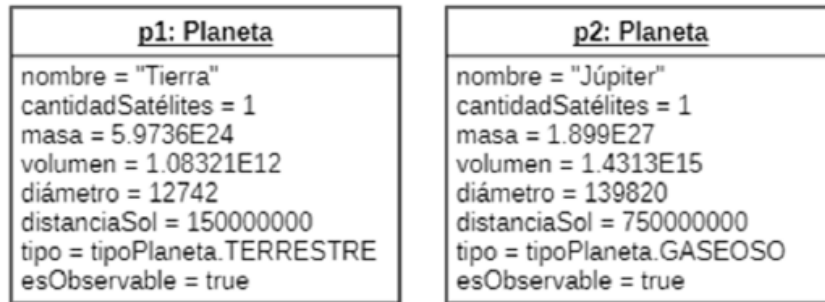
En un método main se deben crear dos planetas y mostrar los valores de sus atributos en pantalla. Además, se debe imprimir la densidad de cada planeta y si el planeta es un planeta exterior del sistema solar.

**Solución:**

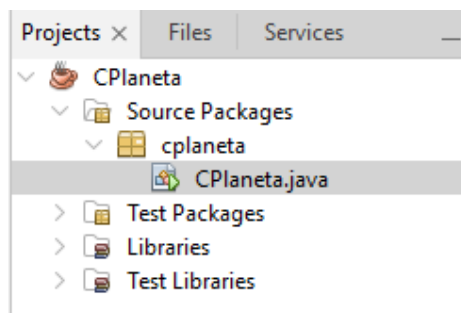
• **Diagrama de clases**



- Diagrama de objetos



- Proyecto



- Código fuente:

```

1 package cplaneta;
2
3 public class CPlaneta {
4
5     String nombre = null;
6     int cantidadSatelites = 0;
7     double masa = 0;
8     double volumen = 0;
9     int diametro = 0;
10    int distanciaSol = 0;
11    enum tipoPlaneta {GASEOSO, TERRESTRE, ENANO}
12    tipoPlaneta tipo;
13    boolean esObservable = false;
14
15    //Constructor
16    CPlaneta(String nombre, int cantidadSatelites, double masa, double volumen,
17            int diametro, int distanciaSol, tipoPlaneta tipo, boolean esObservable){
18        this.nombre = nombre;
19        this.cantidadSatelites = cantidadSatelites;
20        this.masa = masa;
21        this.volumen = volumen;
22        this.diametro = diametro;
    }

```

```

23         this.distanciaSol = distanciaSol;
24         this.tipo = tipo;
25         this.esObservable = esObservable;
26     }
27
28     //Método que imprime en pantalla los datos de un planeta
29     void imprimir() {
30         System.out.println("Nombre del planeta = " + nombre);
31         System.out.println("Cantidad de satélites = " + cantidadSatelites);
32         System.out.println("Masa del planeta = " + masa);
33         System.out.println("Volumen del planeta = " + volumen);
34         System.out.println("Diámetro del planeta = " + diametro);
35         System.out.println("Distancia al sol = " + distanciaSol);
36         System.out.println("Tipo de planeta = " + tipo);
37         System.out.println("Es observable = " + esObservable);
38     }
39
40     //Método que calcula y devuelve la densidad de un planeta
41     double calcularDensidad() {
42         return masa/volumen;
43     }
44
45     //Método que determina y devuelve si un planeta es exterior o no
46     boolean esPlanetaExterior() {
47         float limite = (float) (149597870 * 3.4);
48         if(distanciaSol > limite){
49             return true;
50         }else{
51             return false;
52         }
53     }
54
55     public static void main(String[] args) {
56         CPlaneta p1 = new CPlaneta("Tierra",1,5.9736E24,1.08321E12,12742,150000000,
57             tipoPlaneta.TERRESTRE,true);
58         p1.imprimir();
59         System.out.println("Densidad del planeta = " + p1.calcularDensidad());
60         System.out.println("Es planeta exterior = " + p1.esPlanetaExterior());
61         System.out.println();
62         CPlaneta p2 = new CPlaneta("Júpiter",79,1899E27,1.4313E15,139820,
63             750000000,tipoPlaneta.GASEOSO,true);
64         p2.imprimir();
65         System.out.println("Densidad del planeta = " + p2.calcularDensidad());
66         System.out.println("Es planeta exterior = " + p2.esPlanetaExterior());
67     }
68 }
69

```

**Resultado:**

```
Output - CPlaneta (run) x CPlaneta.java x
run:
Nombre del planeta = Tierra
Cantidad de satélites = 1
Masa del planeta = 5.9736E24
Volumen del planeta = 1.08321E12
Diámetro del planeta = 12742
Distancia al sol = 150000000
Tipo de planeta = TERRESTRE
Es observable = true
Densidad del planeta = 5.514720137369484E12
Es planeta exterior = false

Nombre del planeta = Júpiter
Cantidad de satélites = 79
Masa del planeta = 1.899E30
Volumen del planeta = 1.4313E15
Diámetro del planeta = 139820
Distancia al sol = 750000000
Tipo de planeta = GASEOSO
Es observable = true
Densidad del planeta = 1.3267658771745965E15
Es planeta exterior = true
BUILD SUCCESSFUL (total time: 2 seconds)
```

### 3. ENUNCIADO: Clase Automovil

Se requiere un programa que modele el concepto de un automóvil. Un automóvil tiene los siguientes atributos:

- Marca: el nombre del fabricante.
- Modelo: año de fabricación.
- Motor: volumen en litros del cilindraje del motor de un automóvil.
- Tipo de combustible: valor enumerado con los posibles valores de gasolina, bioetanol, diésel, biodiésel, gas natural.
- Tipo de automóvil: valor enumerado con los posibles valores de carro de ciudad, subcompacto, compacto, familiar, ejecutivo, SUV.
- Número de puertas: cantidad de puertas.
- Cantidad de asientos: número de asientos disponibles que tiene el vehículo.
- Velocidad máxima: velocidad máxima sostenida por el vehículo en km/h.
- Color: valor enumerado con los posibles valores de blanco, negro, rojo, naranja, amarillo, verde, azul, violeta.
- Velocidad actual: velocidad del vehículo en un momento dado.

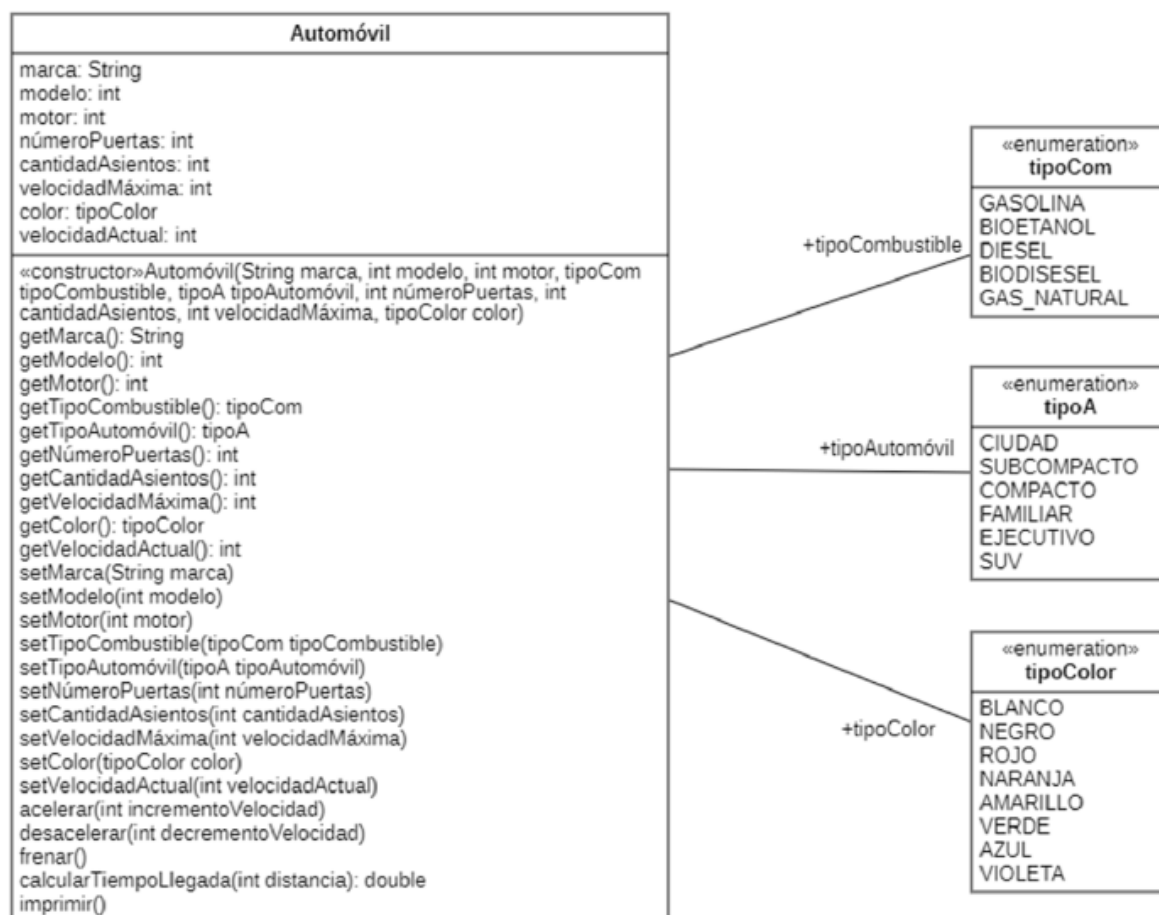
La clase debe incluir los siguientes métodos:

- Un constructor para la clase Automóvil donde se le pasen como parámetros los valores de sus atributos.
- Métodos get y set para la clase Automóvil.

- Métodos para acelerar una cierta velocidad, desacelerar y frenar (colocar la velocidad actual en cero). Es importante tener en cuenta que no se debe acelerar más allá de la velocidad máxima permitida para el automóvil. De igual manera, tampoco es posible desacelerar a una velocidad negativa. Si se cumplen estos casos, se debe mostrar por pantalla los mensajes correspondientes.
- Un método para calcular el tiempo estimado de llegada, utilizando como parámetro la distancia a recorrer en kilómetros. El tiempo estimado se calcula como el cociente entre la distancia a recorrer y la velocidad actual.
- Un método para mostrar los valores de los atributos de un Auto- móvil en pantalla.
- Un método main donde se deben crear un automóvil, colocar su velocidad actual en 100 km/h, aumentar su velocidad en 20 km/h, luego decrementar su velocidad en 50 km/h, y después frenar. Con cada cambio de velocidad, se debe mostrar en pantalla la velocidad actual.

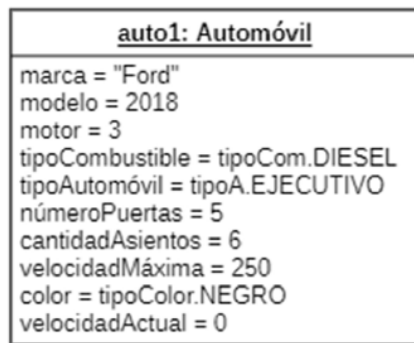
## Solución:

### • Diagrama de clases

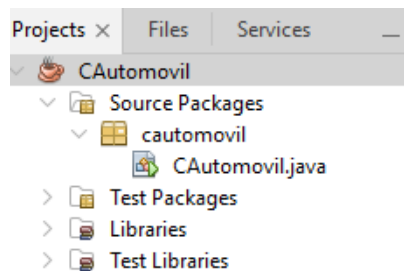




- Diagrama de objetos



- Proyecto



- Código fuente:

```

1 package cautomovil;
2
3 public class CAutomovil {
4
5     String marca;
6     int modelo;
7     int motor;
8     enum tipoCom {GASOLINA, BIOETANOL, DIESEL, BIODIESEL, GAS_NATURAL}
9     tipoCom tipoCombustible;
10    enum tipoA {CIUDAD, SUBCOMPACTO, COMPACTO, FAMILIAR, EJECUTIVO, SUV}
11    tipoA tipoAutomovil;
12    int numeroPuertas;
13    int cantidadAsientos;
14    int velocidadMaxima;
15    enum tipoColor {BLANCO, NEGRO, ROJO, NARANJA, AMARILLO, VERDE, AZUL, VIOLETA}
16    tipoColor color;
17    int velocidadActual = 0;
18
19    //Constructor
20    CAutomovil(String marca, int modelo, int motor, tipoCom tipoCombustible,
21                tipoA tipoAutomovil, int numeroPuertas, int cantidadAsientos,
22                int velocidadMaxima, tipoColor color){
23        this.marca = marca;
24        this.modelo = modelo;
25        this.motor = motor;
26        this.tipoCombustible = tipoCombustible;
27        this.tipoAutomovil = tipoAutomovil;
28        this.numeroPuertas = numeroPuertas;
29        this.cantidadAsientos = cantidadAsientos;
30        this.velocidadMaxima = velocidadMaxima;
31        this.color = color;
32    }

```

```

33
34 //Método que devuelve la marca del automovil
35 String getMarca(){
36     return marca;
37 }
38 //Método que devuelve el modelo de un automovil
39 int getModelo(){
40     return modelo;
41 }
42 //Método que devuelve el volumen en litros del cilindraje del motor
43 int getMotor(){
44     return motor;
45 }
46 //Método que devuelve el tipo de combustible utilizado por el motor
47 tipoCom getTipoCombustible(){
48     return tipoCombustible;
49 }
50 //Método que devuelve el tipo de automovil
51 tipoA getTipoAutomovil(){
52     return tipoAutomovil;
53 }
54 //Método que devuelve el numero de puertas de un automovil
55 int getNumeroPuertas(){
56     return numeroPuertas;
57 }
58 //Método que devuelve la cantidad de asientos de un automovil
59 int getCantidadAsientos(){
60     return cantidadAsientos;
61 }
62 //Método que devuelve la velocidad máxima de un automovil
63 int getVelocidadMaxima(){
64     return velocidadMaxima;
65 }
66 //Método que devuelve el color de un automovil
67 tipoColor getColor(){
68     return color;
69 }
70 //Método que devuelve la velocidad actual de un automovil
71 int getVelocidadActual(){
72     return velocidadActual;
73 }
74 //Método que establece la marca de un automovil
75 void setMarca(String marca){
76     this.marca = marca;
77 }
78 //Método que establece el modelo de un automovil
79 void setModelo(int modelo){
80     this.modelo = modelo;
81 }
82 //Método que establece el volumen en litros del motor de un automovil
83 void setMotor(int motor){
84     this.motor = motor;
85 }
86 //Método que establece el tipo de combustible de un automovil
87 void setTipoCombustible(tipoCom tipoCombustible){
88     this.tipoCombustible = tipoCombustible;
89 }

```

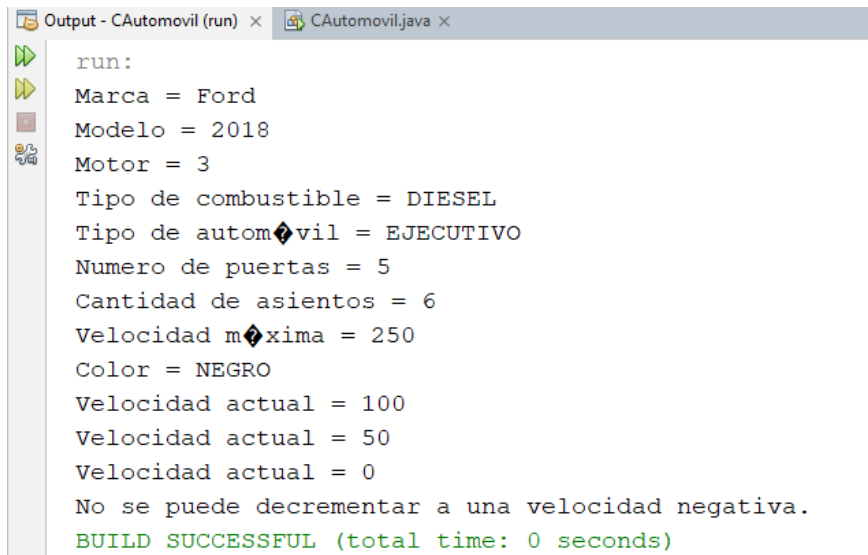
```

90 //Método que establece el tipo de automovil
91 void setTipoAutomovil(tipoA tipoAutomovil){
92     this.tipoAutomovil = tipoAutomovil;
93 }
94 //Método que establece el numero de puertas de un automovil
95 void setNumeroPuertas(int numeroPuertas){
96     this.numeroPuertas = numeroPuertas;
97 }
98 //Método que establece la cantidad de asientos de un automovil
99 void setCantidadAsientos(int cantidadAsientos){
100     this.cantidadAsientos = cantidadAsientos;
101 }
102 //Método que establece la velocidad maxima de un automvil
103 void setVelocidadMaxima(int velocidadMaxima){
104     this.velocidadMaxima = velocidadMaxima;
105 }
106 //Método que estable el color de un automovil
107 void setColor(tipoColor color) {
108     this.color = color;
109 }
110 //Método que establece la velocidad de un automovil
111 void setVelocidadActual(int velocidadActual){
112     this.velocidadActual = velocidadActual;
113 }
114 //Método que incrementa la velocidad de un automovil
115 void acelerar(int incrementoVelocidad) {
116     if(velocidadActual + incrementoVelocidad < velocidadMaxima){
117         //Si el incremento de la velocidad no supera la velocidad maxima
118         velocidadActual = velocidadActual + incrementoVelocidad;
119     }else{
120         System.out.println("No se puede incrementar a una velocidad superior " +
121             "a la maxima del automovil");
122     }
123 }
124 // Método que decrementa la velocidad de un automovil
125 void desacelerar(int decrementoVelocidad){
126     //La velocidad actual no se puede decrementar alcanzando un valor negativo
127     if((velocidadActual - decrementoVelocidad) > 0){
128         velocidadActual = velocidadActual - decrementoVelocidad;
129     }else{
130         System.out.println("No se puede decrementar a una velocidad negativa.");
131     }
132 }
133 //Método que coloca la velocidad actual de un automovil en cero
134 void frenar() {
135     velocidadActual = 0;
136 }
137 //Método que calcula el tiempo que tarda un automovil en recorrer cierta distancia
138 double calcularTiempoLlegada(int distancia){
139     return distancia/velocidadActual;
140 }
141 //Método que imprime en pantalla los valores de los atributos de un automovil.
142 void imprimir() {
143     System.out.println("Marca = " + marca);
144     System.out.println("Modelo = " + modelo);
145     System.out.println("Motor = " + motor);
146     System.out.println("Tipo de combustible = " + tipoCombustible);
147     System.out.println("Tipo de automóvil = " + tipoAutomovil);
148     System.out.println("Numero de puertas = " + numeroPuertas);
149     System.out.println("Cantidad de asientos = " + cantidadAsientos);
150     System.out.println("Velocidad máxima = " + velocidadMaxima);
151     System.out.println("Color = " + color);
152 }

```

```
153  
154 public static void main(String[] args) {  
155     CAutomovil auto1 = new CAutomovil("Ford",2018,3, tipoCom.DIESEL, tipoA.EJECUTIVO,  
156         5,6,250, tipoColor.NEGRO);  
157     auto1.imprimir();  
158     auto1.setVelocidadActual(100);  
159     System.out.println("Velocidad actual = " + auto1.velocidadActual);  
160     auto1.desacelerar(50);  
161     System.out.println("Velocidad actual = " + auto1.velocidadActual);  
162     auto1.frenar();  
163     System.out.println("Velocidad actual = " + auto1.velocidadActual);  
164     auto1.desacelerar(20);  
165 }  
166 }  
167
```

## Resultado



```
Output - CAutomovil (run) x CAutomovil.java x  
run:  
Marca = Ford  
Modelo = 2018  
Motor = 3  
Tipo de combustible = DIESEL  
Tipo de automovil = EJECUTIVO  
Numero de puertas = 5  
Cantidad de asientos = 6  
Velocidad maxima = 250  
Color = NEGRO  
Velocidad actual = 100  
Velocidad actual = 50  
Velocidad actual = 0  
No se puede decrementar a una velocidad negativa.  
BUILD SUCCESSFUL (total time: 0 seconds)
```

## EJERCICIO PROPUESTO

- Agregar a la clase Automóvil, un atributo para determinar si el vehículo es automático o no. Agregar los métodos get y set para dicho atributo. Modificar el constructor para inicializar dicho atributo.
- Modificar el método acelerar para que si la velocidad máxima se sobrepase se genere una multa. Dicha multa se puede incrementar cada vez que el vehículo intenta superar la velocidad máxima permitida.
- Agregar un método para determinar si un vehículo tiene multas y otro método para determinar el valor total de multas de un vehículo.