ADRIAN MAXIMILIANO MUNOZ MARTINEZ

Laboratorio POO  Sabado 7 am – 9am

PRactica 9: Java COllection

**Introducción**

Se desarrollará una aplicación en Java que en la que se implemente colecciones

**Objetivo**

El objetivo de la practica es familiarizarnos con el concepto de collection, así como practicar su implementación y conocer sus ventajas y desventajas

**Practica**

Practica9.java

package practica9;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

public class Practica9 {

public static void main(String[] args) {

// Inicializar scanner

Scanner sc = new Scanner(System.in);

// Crear array list

ArrayList autos = new ArrayList();

// Ciclo for en el que se pedira la informacion de los 10 autos que seran

// agregados a la lista

for (int i = 0; i < 10; i++) {

// Darle formato al String para incluir, el nuumero del auto que sera anadido

System.out.print(String.format("\nAutomovil numero %d", i + 1));

// Llamar a constructor del objeto Automovil

Automovil auto = new Automovil(10);

// Llenar los atributos del objeto, con los vaolres indicados por el usuario

System.out.print("\nIngresa la marca del automvil: ");

auto.setMarca(sc.nextLine());

System.out.print("Ingresa el modelo del automovil: ");

auto.setModelo(sc.nextLine());

System.out.print("Ingresa el precio del automovil: ");

auto.setPrecio(sc.nextLine());

// Agregar nuevo objeto a la lista

autos.add(auto);

}

// Imprimir los atributos y llamar las funciones de cada objeto Automovil en el

// array list

for (Object objectAuto : autos) {

// Convertir el elemento del array a tipo auto, y guardar la referencia en una

// nueva variable

// De esta manera Java va a saber de que tipo de objeto estamos hablando

Automovil auto = (Automovil) objectAuto;

// Imprimir el numero del automovil en el orden en que se ingreso

System.out.println(String.format("\nAutomovil %d", (autos.indexOf(auto) + 1)));

// Imprimir los atributos y llamar a los metodos de la clase Automovil

System.out.println("Marca: " + auto.getMarca());

System.out.println("Modelo: " + auto.getModelo());

System.out.println("Precio: " + auto.getPrecio());

// Preguntar al usario por la direccion en que desea moverse

System.out.println("Ingresa la direccion en la que el auto tiene que moverse: ");

System.out.println(auto.movimiento(sc.nextLine()));

System.out.println(auto.alto());

}

}

}

Automovil.java

package practica9;

public class Automovil {

private String marca;

private String modelo;

private String precio;

private int gas;

// Constructor

public Automovil(int gas) {

this.encender(gas);

}

// Encender funciona como un setter de gas

private void encender(int gas) {

this.gas = gas;

}

public String alto() {

return "El automovil se ha detenido";

}

public String movimiento(String mov) {

if (this.gas > 1) {

return "Moviendose a la " + mov;

} else {

return "No hay suficiente gasolina";

}

}

public void setMarca(String marca) {

this.marca = marca;

}

public void setModelo(String modelo) {

this.modelo = modelo;

}

public void setPrecio(String precio) {

this.precio = precio;

}

public String getMarca() {

return this.marca;

}

public String getModelo() {

return this.modelo;

}

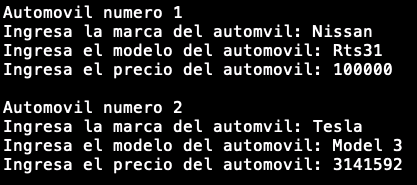
public String getPrecio() {

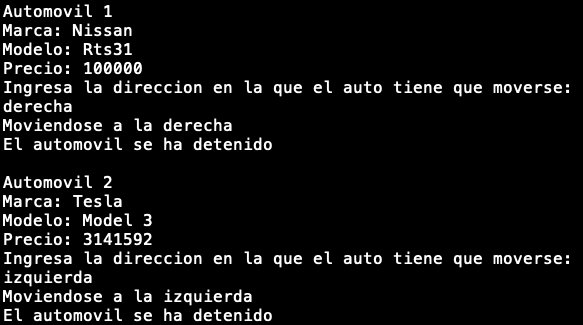
return this.precio;

}

}

**Screenshots**



****

**Explicación**

Se creo una aplicación que implemento un objeto Automóvil. Dentro de la clase principal (Practica9) se creo un ArrayList cuya función era la de almacenar los objetos de tipo Automovil que fueron creados y configurados con la entrada del usuario.

Para acceder a esos objetos fue necesario castear cada uno de los elementos del ArrayList y guardar la referencia del objeto casteado en una variable de tipo Automovil. Se hizo esto ya que Java no sabia que tipo de elementos se encontraban en el arreglo y por lo tanto no nos permitía acceder a las funciones de la clase Automovil, porque hasta donde Java sabia dentro del ArrayList solo había objetos, pero no sabia de que tipo.

Por eso fue necesario castear y almacenar el resultado en una variable tipo Automovil, para que Java supiera que tipo de objeto era determinado elemento del ArrayList y pudiéramos acceder a las funciones de Automovil.

**Conclusión**

Las colecciones en Java son muy útiles ya que nos permiten almacenar información y objetos de una manera indeterminada, ya sea en tipo o numero. También ofrecen funciones que nos permiten manipular y obtener la información dentro del array muy fácilmente, siempre y cuando le especifiquemos el tipo de dato que se esta utilizando.