# 07. Programación Orientada a Objetos (POO)

**Padilla Herrera Carlos Ignacio**

**Folio: 794DR02**

**Escuela de Código PILARES**
**Java Entre Semana G0224**

Fecha: 03 de junio de 2024

# 07. Programación Orientada a Objetos (POO)

Carlos Ignacio Padilla Herrera

03 de junio de 2024

## 07. Programación Orientada a Objetos (POO)

**Objetivo:** Verificar el dominio teórico y técnico del paradigma programación orientada a objetos mediante ejercicios para desarrollar su código y preguntas de respuesta múltiple.

**Indicaciones:** Realiza los programas que se te solicitan, una vez realizado sube el archivo de tu actividad.

## Actividades

### Programa 1. (3 puntos)

Crea un programa que forme parte de un módulo para el sistema de una estética de perros. El programa deberá de contener lo siguiente:

- Una clase *RecibeMascota*.
- Los atributos: *nombre de perro*, *edad*, *raza*, *tamaño* y *nombre de dueño*.
- Crear el método *main* en donde contendrá el código para realizar el siguiente procedimiento:
  1. Crear un arreglo dinámico de tipo de la clase.
  2. Hacer una instancia para crear un objeto de tipo de la clase.
  3. Asignar valores a los atributos de la clase con datos que tú prefieras.
  4. Agregar el objeto creado al arreglo.
  5. Imprimir la cantidad de perros que se encuentran en la estética con el siguiente mensaje: *Perros actuales en la estética: ¡Numero de perros¿*.

Toma captura de pantalla del código completo y del programa compilado.

```
1
2  package com.mycompany.recibemascota;
3
4  import java.util.ArrayList;
5
6  public class RecibeMascota {
7
8     String nombrePerro;
9     int edad;
10    String raza;
11    String tamanio;
12    String nombreDuenio;
13
14    public RecibeMascota(String nombrePerro, int edad, String raza, String tamanio, String
          nombreDuenio) {
15       this.nombrePerro = nombrePerro;
16       this.edad = edad;
17       this.raza = raza;
18       this.tamanio = tamanio;
19       this.nombreDuenio = nombreDuenio;
20    }
21
22    public static void main(String[] args) {
23       // a. Crear un arreglo dinamico de tipo de la clase
24       ArrayList<RecibeMascota> listaMascotas = new ArrayList<>();
25
26       // b. Hacer instancias para crear objetos de tipo de la clase
27       RecibeMascota mascota1 = new RecibeMascota("Firulais", 3, "Labrador", "Grande", "Juan Perez
             ");
28       RecibeMascota mascota2 = new RecibeMascota("Tyzon", 2, "Bulldog", "Mediano", "Ana Garcia");
29       RecibeMascota mascota3 = new RecibeMascota("Luna", 1, "Chihuahua", "Pequenio", "Luis
             Martinez");
30       RecibeMascota mascota4 = new RecibeMascota("Rocky", 4, "Pastor Aleman", "Grande", "Marta
             Lopez");
31       RecibeMascota mascota5 = new RecibeMascota("Max", 3, "Golden Retriever", "Grande", "Pedro
             Hernandez");
32       RecibeMascota mascota6 = new RecibeMascota("Bella", 5, "Beagle", "Mediano", "Sofia Gonzalez
             ");
33       RecibeMascota mascota7 = new RecibeMascota("Lucky", 2, "Poodle", "Pequenio", "Carlos
             Ramirez");
34
35       // c. Agregar los objetos creados al arreglo
36       listaMascotas.add(mascota1);
37       listaMascotas.add(mascota2);
38       listaMascotas.add(mascota3);
39       listaMascotas.add(mascota4);
40       listaMascotas.add(mascota5);
41       listaMascotas.add(mascota6);
42       listaMascotas.add(mascota7);
43
44       // d. Imprimir la cantidad de perros que se encuentran en la esttica
45       System.out.println("Perros actuales en la estetica: " + listaMascotas.size());
46    }
47 }
```

Listing 1: Código base del programa RecibeMascota.java

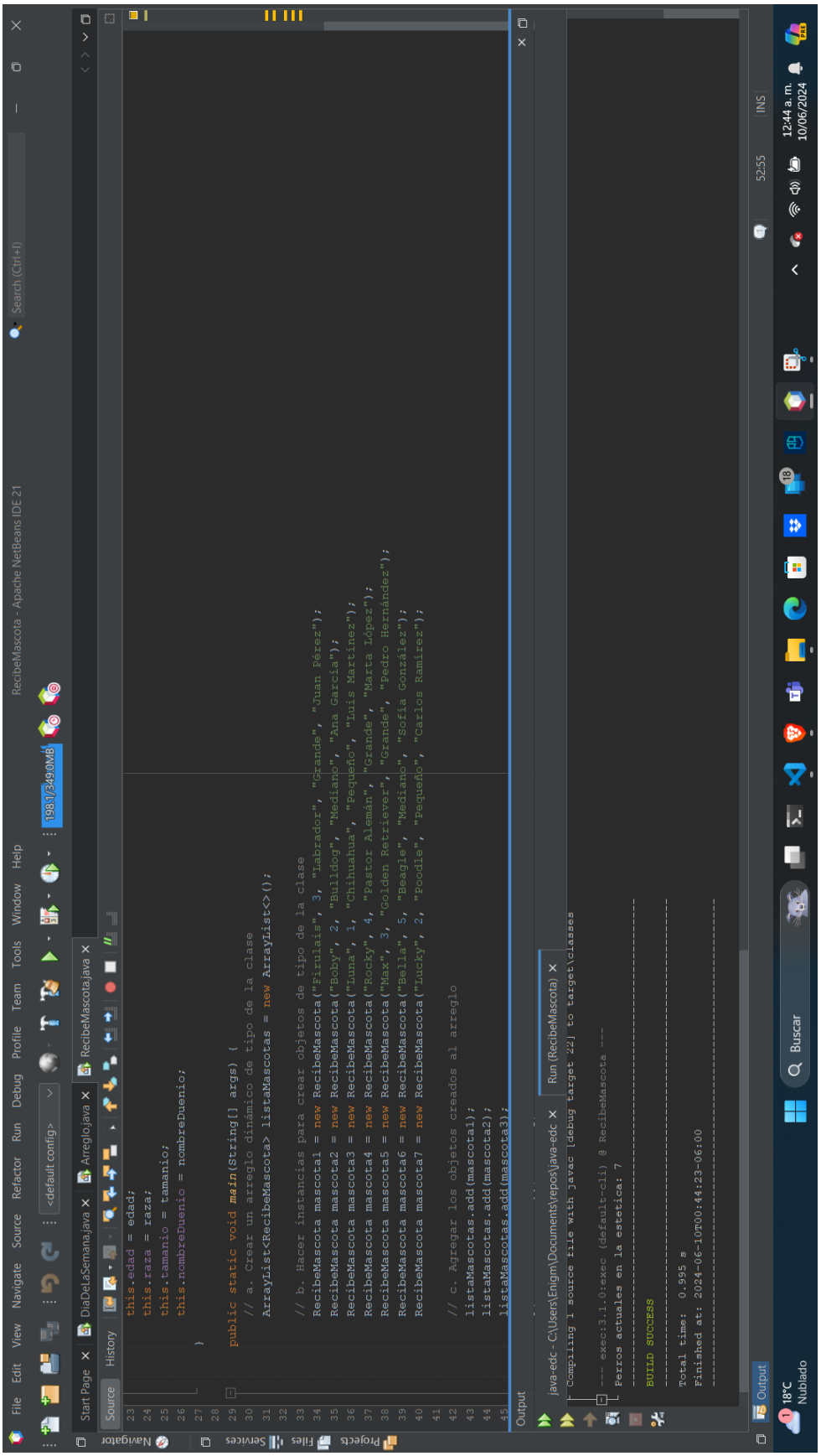


Figure 1: Captura del programa RecibeMascota.java