

PROGRAMACIÓN II

Trabajo Práctico 7: Herencia y Polimorfismo en Java

Alumno :

- González Federico

Profesor :

- Enferrel Ariel

Tutor :

- Ferro Tomás

GitHub:

- <https://github.com/fedeglz/UTN-TUPaD-P2.git>

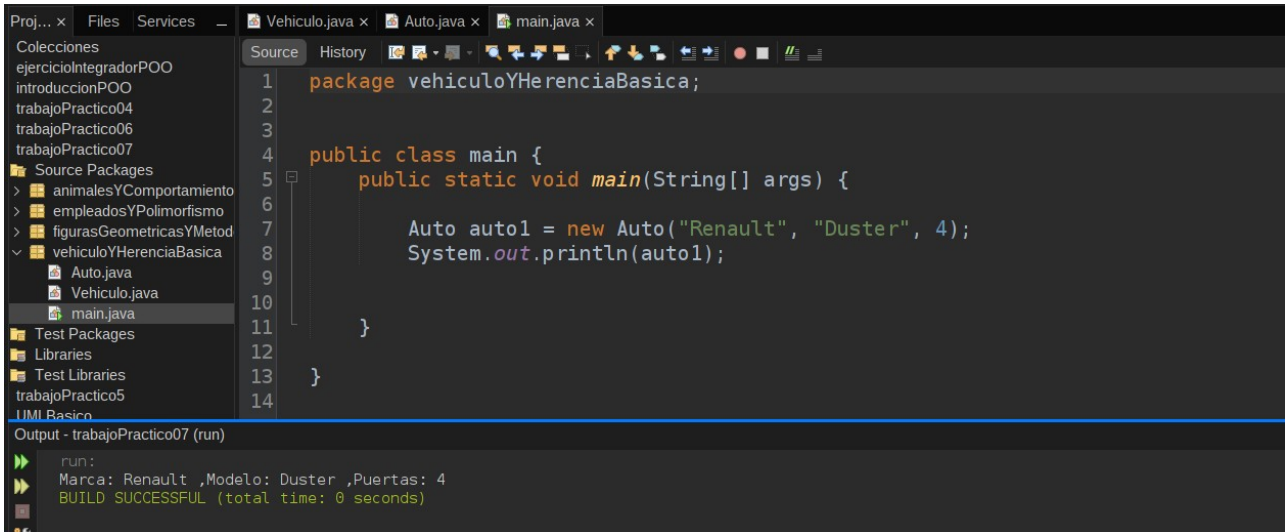
1. Vehículos y herencia básica

- Clase base: Vehículo con atributos marca, modelo y método mostrarInfo()
- Subclase: Auto con atributo adicional cantidadPuertas, sobrescribe mostrarInfo()
- Tarea: Instanciar un auto y mostrar su información completa.

```
Vehiculo.java x
Source History
1 package vehiculoYHerenciaBasica;
2
3
4
5 public class Vehiculo {
6     protected String marca;
7     protected String modelo;
8
9     public Vehiculo(String marca, String modelo) {
10         this.marca = marca;
11         this.modelo = modelo;
12     }
13
14     public String mostrarInfo(){
15         return "Marca: " + marca + " ,Modelo: " + modelo;
16     }
17
18 }
19
```

```
Vehiculo.java x Auto.java x
Source History
1 package vehiculoYHerenciaBasica;
2
3
4 class Auto extends Vehiculo{
5
6     private int cantidadPuertas;
7
8     public Auto(String marca, String modelo, int cantidadPuertas) {
9         super(marca, modelo);
10         this.cantidadPuertas = cantidadPuertas;
11     }
12
13     @Override
14     public String toString() {
15         return super.mostrarInfo() + " ,Puertas: " + cantidadPuertas;
16     }
17
18 }
19
20
21
```

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA



```

1 package vehiculoYHerenciaBasica;
2
3
4 public class main {
5     public static void main(String[] args) {
6
7         Auto auto1 = new Auto("Renault", "Duster", 4);
8         System.out.println(auto1);
9
10    }
11
12 }
13
14

```

Output - trabajoPractico07 (run)

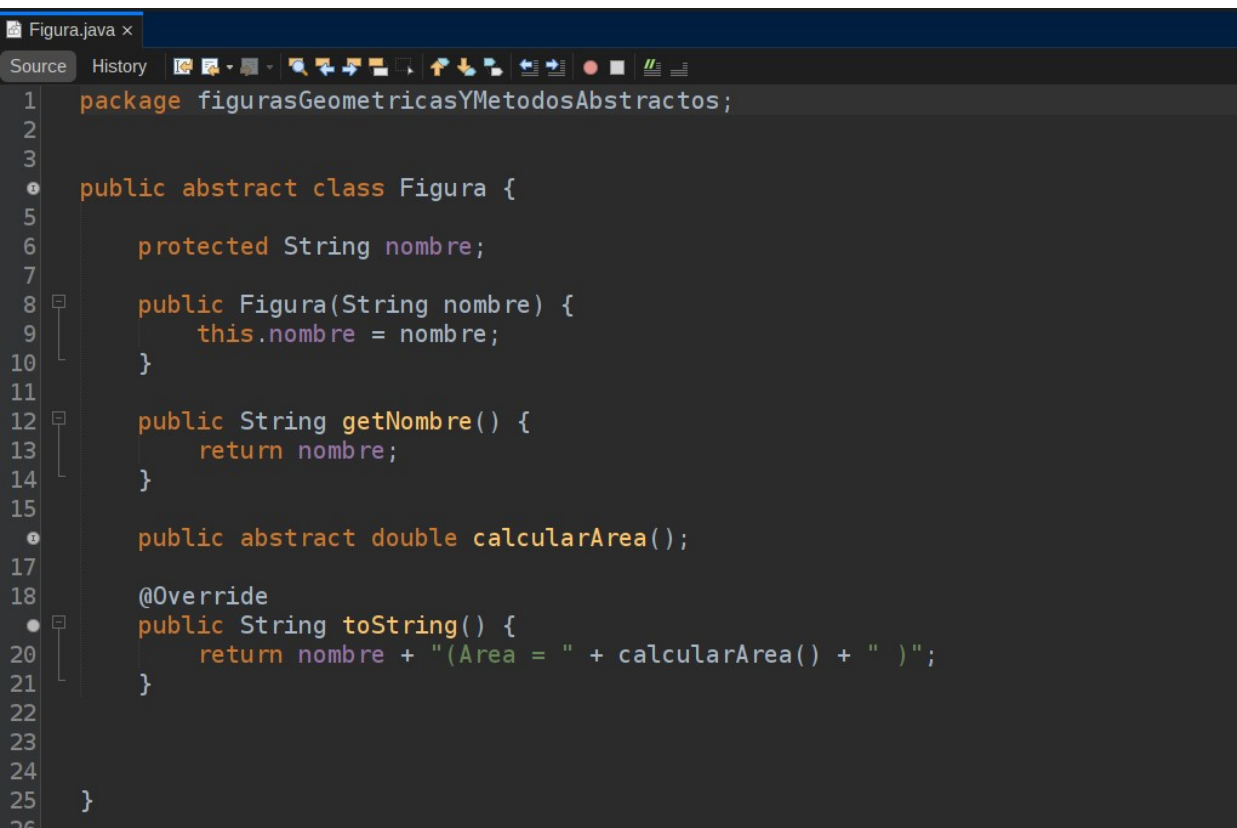
```

run:
Marca: Renault ,Modelo: Duster ,Puertas: 4
BUILD SUCCESSFUL (total time: 0 seconds)

```

2. Figuras geométricas y métodos abstractos

- Clase abstracta: Figura con método calcularArea() y atributo nombre
- Subclases: Círculo y Rectángulo implementan el cálculo del área
- Tarea: Crear un array de figuras y mostrar el área de cada una usando polimorfismo.



```

1 package figurasGeometricasYMetodosAbstractos;
2
3
4 public abstract class Figura {
5
6     protected String nombre;
7
8     public Figura(String nombre) {
9         this.nombre = nombre;
10    }
11
12     public String getNombre() {
13         return nombre;
14    }
15
16     public abstract double calcularArea();
17
18     @Override
19     public String toString() {
20         return nombre + "(Area = " + calcularArea() + " )";
21    }
22
23
24
25 }
26

```

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

```

1 package figurasGeometricasYMetodosAbstractos;
2
3
4 public class Circulo extends Figura {
5
6     private double radio;
7
8     public Circulo(double radio) {
9         super("Circulo");
10        this.radio = radio;
11    }
12
13    public double getRadio() {
14        return radio;
15    }
16
17    @Override
18    public double calcularArea() {
19        return Math.PI * radio * radio;
20    }
21
22    @Override
23    public String toString() {
24        return super.toString() + " ,Radio = " + radio;
25    }
26
27
28
29 }
30

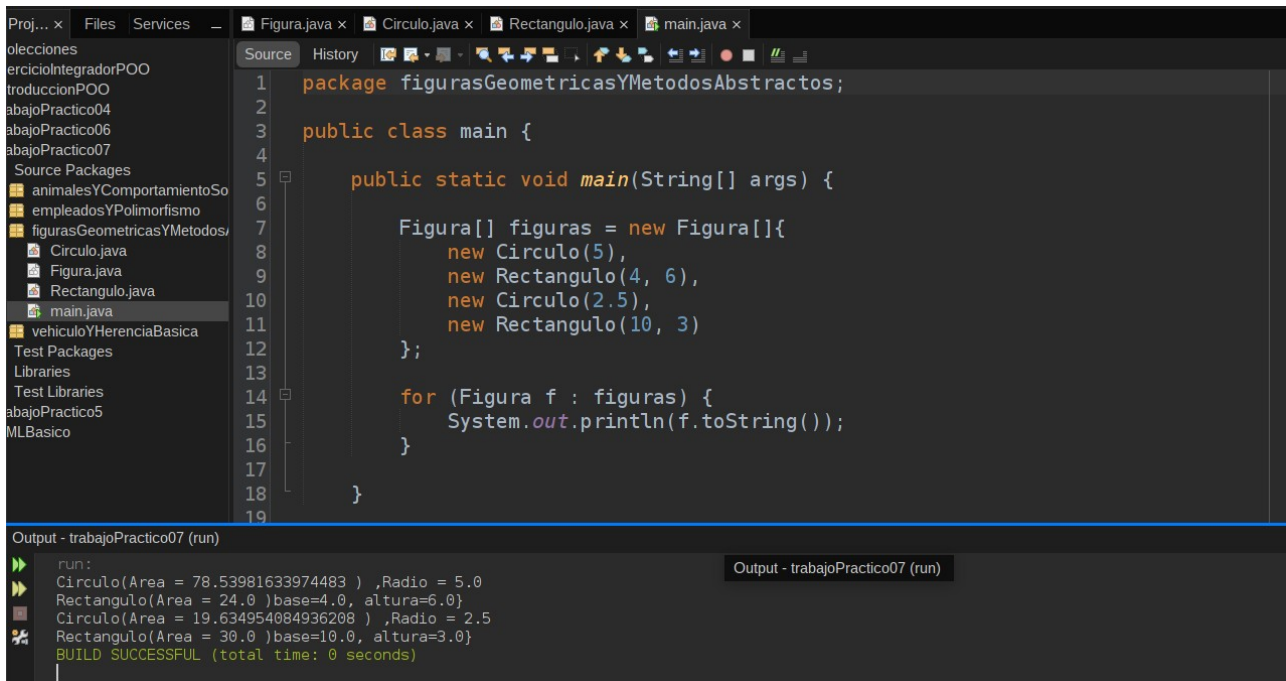
```

```

1 package figurasGeometricasYMetodosAbstractos;
2
3
4 public class Rectangulo extends Figura {
5
6     private double base;
7     private double altura;
8
9     public Rectangulo(double base, double altura) {
10        super("Rectangulo");
11        this.base = base;
12        this.altura = altura;
13    }
14
15    public double getBase() {
16        return base;
17    }
18
19    public double getAltura() {
20        return altura;
21    }
22
23    public void setBase(double base) {
24        this.base = base;
25    }
26
27    public void setAltura(double altura) {
28        this.altura = altura;
29    }
30
31    @Override
32    public double calcularArea() {
33        return base * altura;
34    }
35
36    @Override
37    public String toString() {
38        return super.toString() + "base=" + base + ", altura=" + altura + ' ';
39    }
40
41
42
43 }
44

```

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA



```

package figurasGeometricasYMetodosAbstractos;

public class main {

    public static void main(String[] args) {

        Figura[] figuras = new Figura[]{
            new Circulo(5),
            new Rectangulo(4, 6),
            new Circulo(2.5),
            new Rectangulo(10, 3)
        };

        for (Figura f : figuras) {
            System.out.println(f.toString());
        }
    }
}

```

Output - trabajoPractico07 (run)

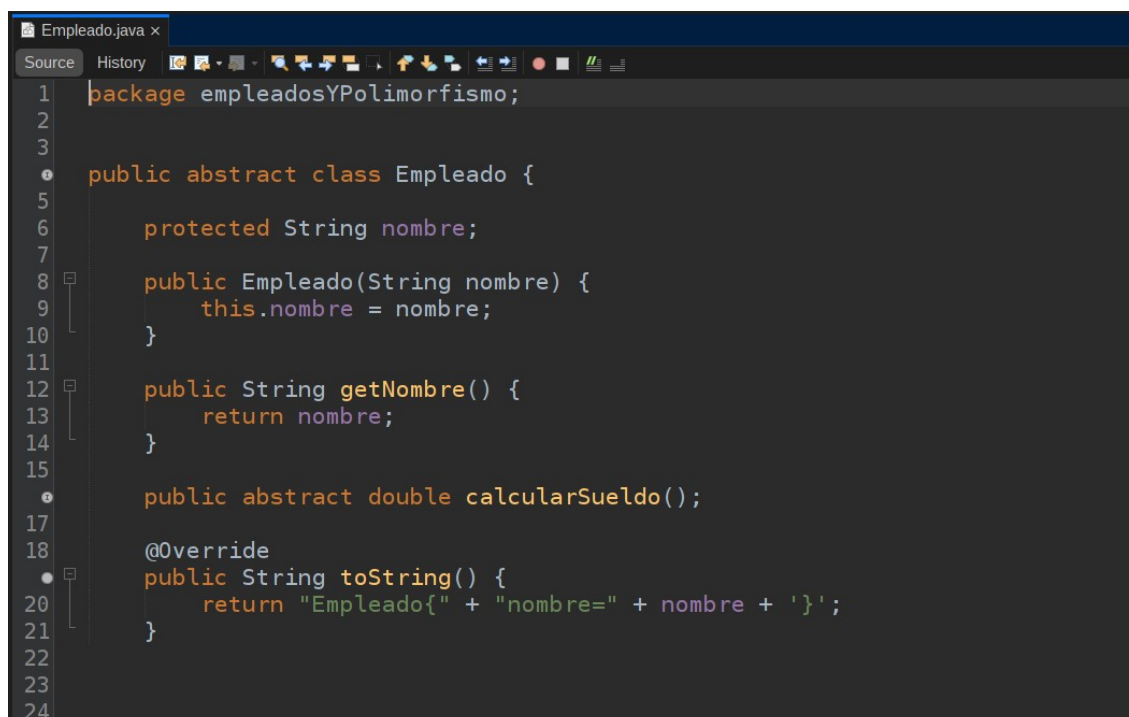
```

run:
Circulo(Area = 78.53981633974483 ) ,Radio = 5.0
Rectangulo(Area = 24.0 )base=4.0, altura=6.0
Circulo(Area = 19.634954084936208 ) ,Radio = 2.5
Rectangulo(Area = 30.0 )base=10.0, altura=3.0
BUILD SUCCESSFUL (total time: 0 seconds)

```

3. Empleados y polimorfismo

- Clase abstracta: Empleado con método calcularSueldo()
- Subclases: EmpleadoPlanta, EmpleadoTemporal
- Tarea: Crear lista de empleados, invocar calcularSueldo() polimórficamente, usar instanceof para clasificar



```

package empleadosYPolimorfismo;

public abstract class Empleado {

    protected String nombre;

    public Empleado(String nombre) {
        this.nombre = nombre;
    }

    public String getNombre() {
        return nombre;
    }

    public abstract double calcularSueldo();

    @Override
    public String toString() {
        return "Empleado{" + "nombre=" + nombre + '}';
    }
}

```


TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA

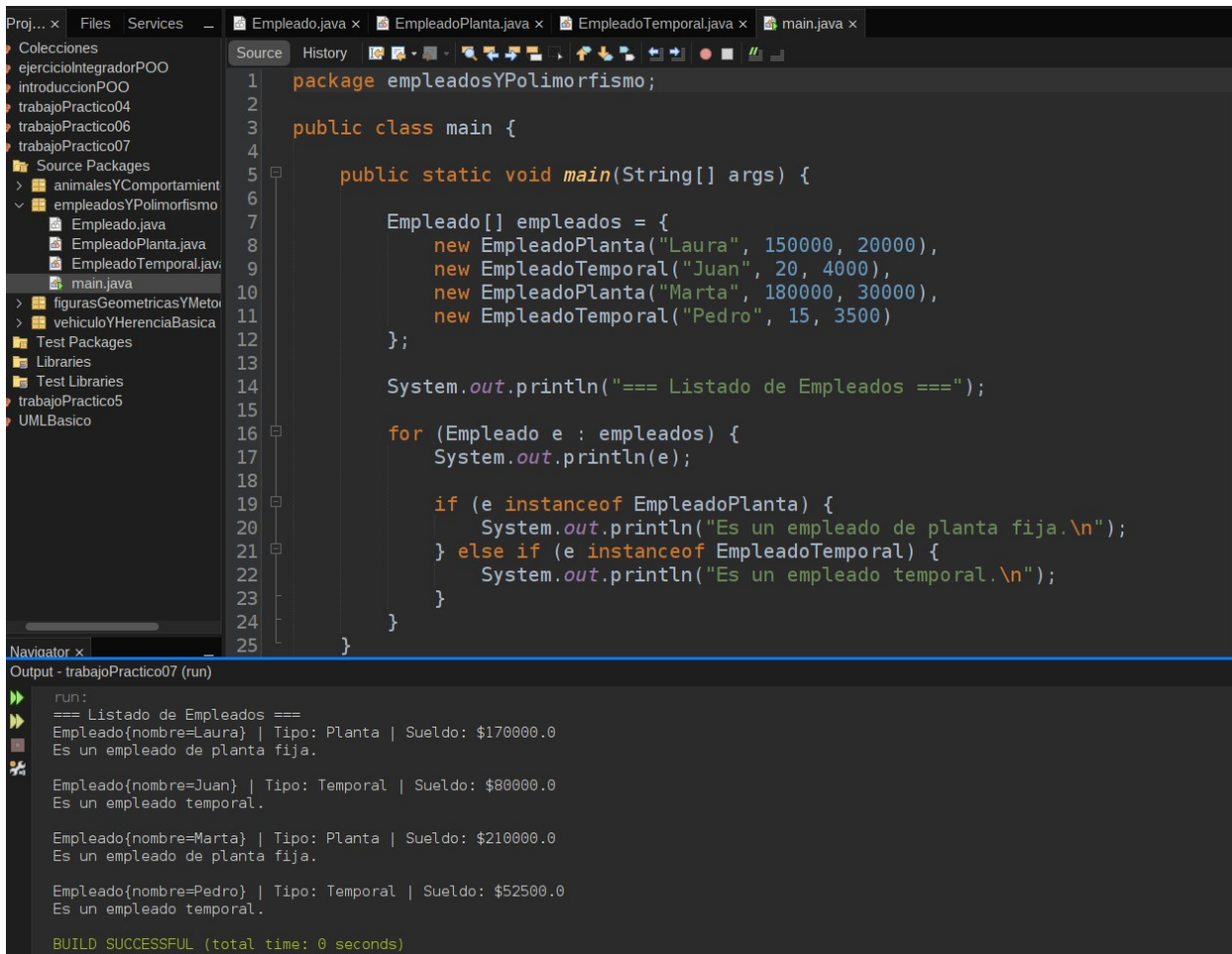
```

Empleado.java x EmpleadoPlanta.java x
Source History
1 package empleadosYPolimorfismo;
2
3
4 public class EmpleadoPlanta extends Empleado {
5
6     private double salarioBase;
7     private double adicionalAntiguedad;
8
9     public EmpleadoPlanta(String nombre, double salarioBase, double adicionalAntiguedad) {
10         super(nombre);
11         this.salarioBase = salarioBase;
12         this.adicionalAntiguedad = adicionalAntiguedad;
13     }
14
15     @Override
16     public double calcularSueldo(){
17         return salarioBase + adicionalAntiguedad;
18     }
19
20     @Override
21     public String toString() {
22         return super.toString() + " | Tipo: Planta | Sueldo: $" + calcularSueldo();
23     }
24
25
26
27
  
```

```

Empleado.java x EmpleadoPlanta.java x EmpleadoTemporal.java x
Source History
1 package empleadosYPolimorfismo;
2
3
4 public class EmpleadoTemporal extends Empleado {
5
6     private int diasTrabajados;
7     private double pagoPorDia;
8
9     public EmpleadoTemporal(String nombre, int diasTrabajados, double pagoPorDia) {
10         super(nombre);
11         this.diasTrabajados = diasTrabajados;
12         this.pagoPorDia = pagoPorDia;
13     }
14
15     @Override
16     public double calcularSueldo(){
17         return diasTrabajados * pagoPorDia;
18     }
19
20     @Override
21     public String toString() {
22         return super.toString() + " | Tipo: Temporal | Sueldo: $" + calcularSueldo();
23     }
24
25
26
27
  
```

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN A DISTANCIA



```

1 package empleadosYPolimorfismo;
2
3 public class main {
4
5     public static void main(String[] args) {
6
7         Empleado[] empleados = {
8             new EmpleadoPlanta("Laura", 150000, 20000),
9             new EmpleadoTemporal("Juan", 20, 4000),
10            new EmpleadoPlanta("Marta", 180000, 30000),
11            new EmpleadoTemporal("Pedro", 15, 3500)
12        };
13
14        System.out.println("=== Listado de Empleados ===");
15
16        for (Empleado e : empleados) {
17            System.out.println(e);
18
19            if (e instanceof EmpleadoPlanta) {
20                System.out.println("Es un empleado de planta fija.\n");
21            } else if (e instanceof EmpleadoTemporal) {
22                System.out.println("Es un empleado temporal.\n");
23            }
24        }
25    }
26 }
  
```

Output - trabajoPractico07 (run)

```

run:
=== Listado de Empleados ===
Empleado{nombre=Laura} | Tipo: Planta | Sueldo: $170000.0
Es un empleado de planta fija.

Empleado{nombre=Juan} | Tipo: Temporal | Sueldo: $80000.0
Es un empleado temporal.

Empleado{nombre=Marta} | Tipo: Planta | Sueldo: $210000.0
Es un empleado de planta fija.

Empleado{nombre=Pedro} | Tipo: Temporal | Sueldo: $52500.0
Es un empleado temporal.

BUILD SUCCESSFUL (total time: 0 seconds)
  
```

4. Animales y comportamiento sobrescrito

- Clase: Animal con método hacerSonido() y describirAnimal()
- Subclases: Perro, Gato, Vaca sobrescriben hacerSonido() con @Override
- Tarea: Crear lista de animales y mostrar sus sonidos con polimorfismo

```
Animal.java x
Source History
1 package animalesYComportamientoSobrescrito;
2
3
4
5 public class Animal {
6     protected String nombre;
7
8
9     public Animal(String nombre) {
10         this.nombre = nombre;
11     }
12
13     public void hacerSonido(){
14         System.out.println("Sonido de animal");
15     }
16
17     public void describirAnimal(){
18         System.out.println("Soy un animal llamado: " + nombre);
19     }
20
21
22
23
24 }
25
```

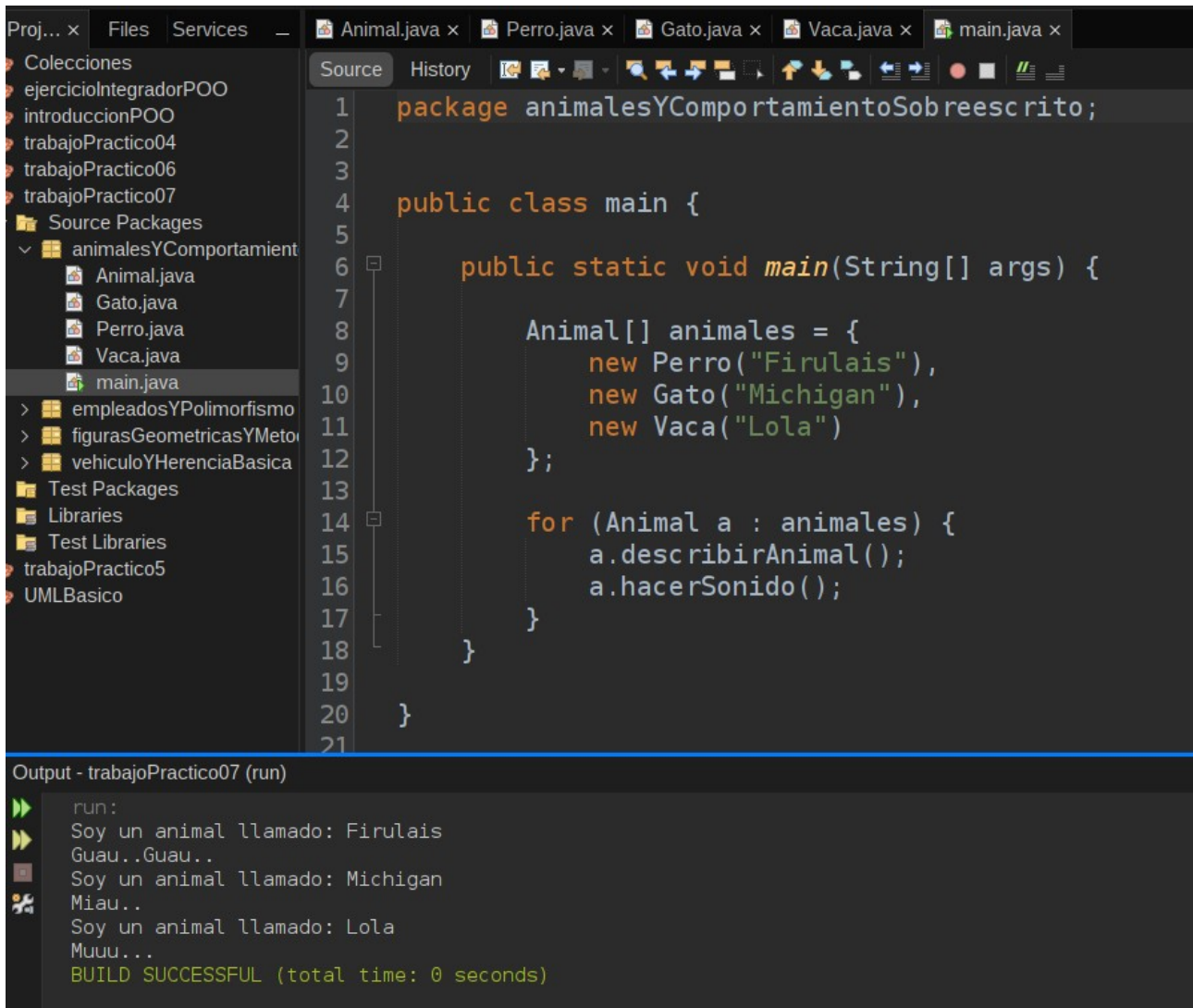
```
Animal.java x Perro.java x
Source History
1 package animalesYComportamientoSobrescrito;
2
3
4 public class Perro extends Animal {
5
6     public Perro(String nombre) {
7         super(nombre);
8     }
9
10    @Override
11    public void hacerSonido(){
12        System.out.println("Guau..Guau..");
13    }
14
15
16 }
17
```


TECNICATURA UNIVERSITARIA EN
PROGRAMACIÓN A DISTANCIA

```
Animal.java x Perro.java x Gato.java x
Source History
1 package animalesYComportamientoSobreescrito;
2
3 public class Gato extends Animal {
4
5
6     public Gato(String nombre) {
7         super(nombre);
8     }
9
10    @Override
11    public void hacerSonido() {
12        System.out.println("Miau..");
13    }
14 }
15
```

```
Animal.java x Perro.java x Gato.java x Vaca.java x
Source History
1 package animalesYComportamientoSobreescrito;
2
3
4 public class Vaca extends Animal {
5
6     public Vaca(String nombre) {
7         super(nombre);
8     }
9
10    @Override
11    public void hacerSonido() {
12        System.out.println("Muuu...");
13    }
14 }
15
16
```

TECNICATURA UNIVERSITARIA EN
PROGRAMACIÓN A DISTANCIA



The screenshot shows an IDE with a project named 'trabajoPractico07'. The left sidebar displays a file explorer with a tree structure including 'Colecciones', 'ejercicioIntegradorPOO', 'introduccionPOO', 'trabajoPractico04', 'trabajoPractico06', 'trabajoPractico07', 'Source Packages', 'animalesYComportamiento' (expanded), 'Animal.java', 'Gato.java', 'Perro.java', 'Vaca.java', 'main.java', 'empleadosYPolimorfismo', 'figurasGeometricasYMetodo', 'vehiculoYHerenciaBasica', 'Test Packages', 'Libraries', 'Test Libraries', 'trabajoPractico5', and 'UMLBasico'. The main editor window shows the code for 'main.java' with the following content:

```
1 package animalesYComportamientoSobreescrito;
2
3
4 public class main {
5
6     public static void main(String[] args) {
7
8         Animal[] animales = {
9             new Perro("Firulais"),
10            new Gato("Michigan"),
11            new Vaca("Lola")
12        };
13
14        for (Animal a : animales) {
15            a.describirAnimal();
16            a.hacerSonido();
17        }
18    }
19
20 }
21
```

Below the code editor, the 'Output' window shows the results of running the program:

```
run:
Soy un animal llamado: Firulais
Guau..Guau..
Soy un animal llamado: Michigan
Miau..
Soy un animal llamado: Lola
Muuu...
BUILD SUCCESSFUL (total time: 0 seconds)
```