

## Hoja de Trabajo 2.1: Superclases y Subclases

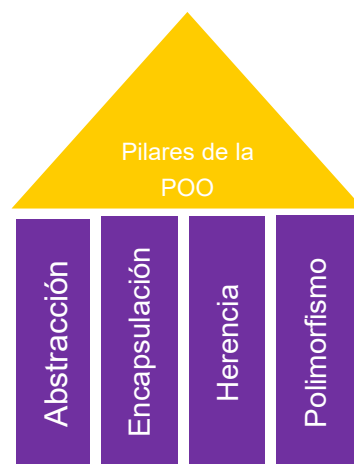
<b>Componente:</b>	Programación de Software II	<b>Grupo:</b>	SI1V
<b>Unidad:</b>	Aplica las características Herencia y Polimorfismo de la Orientación a Objetos	<b>Subgrupo:</b>	Grupo 1 [G1]
<b>Contenido:</b>	Diseña y Utiliza Superclases y Subclases al desarrollar programas de computadoras	<b>Fecha:</b>	17/04/2023
<b>Docente:</b>	Lawdee Norman Narváez B. / Kenet Salinas		

### Fundamentos Teóricos:

A como lo explican Deitel & Deitel (2016), la herencia en la programación orientada a objetos, con la herencia se crea una nueva clase que adquiere los miembros de una clase ya existente, y se mejora con nuevas capacidades o modificando ya las existentes.

La herencia es considerada como uno de los pilares fundamentales de la programación orientada a objetos, en conjunto con la Abstracción, la Encapsulación y el Polimorfismo.

Por medio de la herencia es posible llevar a cabo la reutilización de código, dado que los programadores, a través de las clases existentes pueden ir formando jerarquías de clases dentro de la aplicación. Esto permite ahorrar tiempo en el desarrollo de la aplicación.



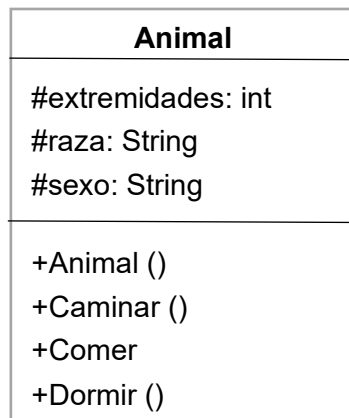
### Términos de consideración

- **Superclases o clase Base:** las clases de las cuales se heredan características (atributos, miembros o campos y métodos) en el mundo de la programación orientada a objetos son conocidas como superclases o clase base.
- **Subclase o clase derivada:** las clases que heredan las características o métodos de otra clase, se conocen en el ámbito de la programación orientada a objetos como subclase, clase hija o bien clase derivada.
- **Reutilización:** por medio de la herencia se aplica el concepto de “reutilización”, cuando se quiere crear una clase, y ya hay una existente que incluye parte del código necesario para la nueva clase, se puede derivar de la clase existente; a lo cual al hacerlo implica que se esta reutilizando los campos y métodos.

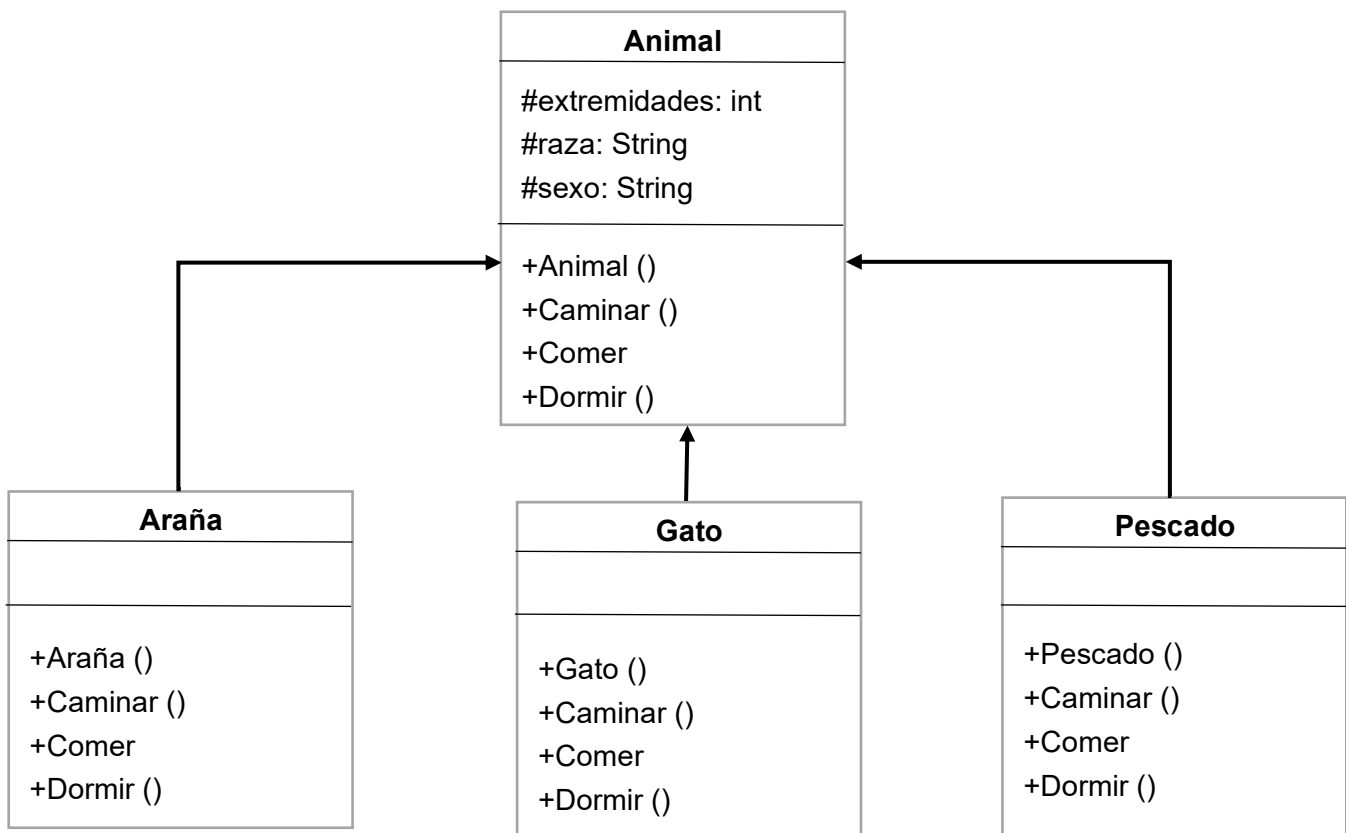
## Representación de herencia en notación UML

La representación de herencia en la orientación a objetos es posible a través de la notación UML; considerando que para representar clases en notación UML ya existe una notación definida. Por tanto, solo hay que utilizar una notación por medio de la cual se lleve a cabo la representación de una relación jerárquica de herencia.

Representación  
de una clase en UML



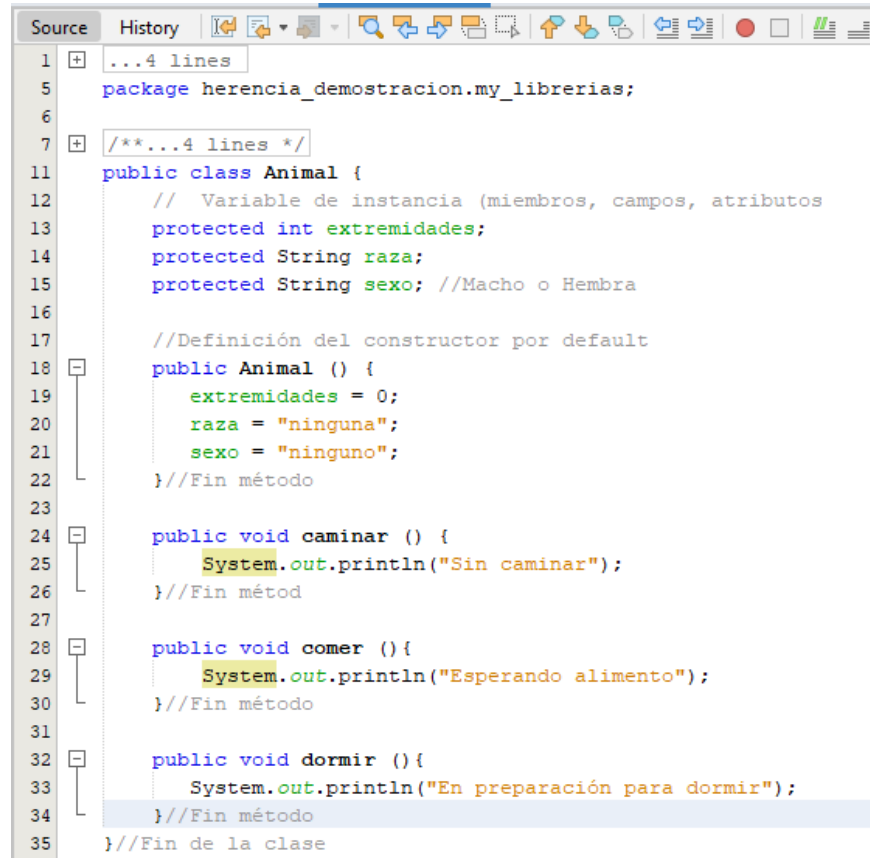
Representación de Herencia (jerarquía de clases en UML)



Para representar que una subclase hereda de una superclase, la punta de la flecha debe apuntar a la superclase y el punto de partida deberá de ser la subclase.

## Demostración de implementación de herencia haciendo uso del lenguaje de programación

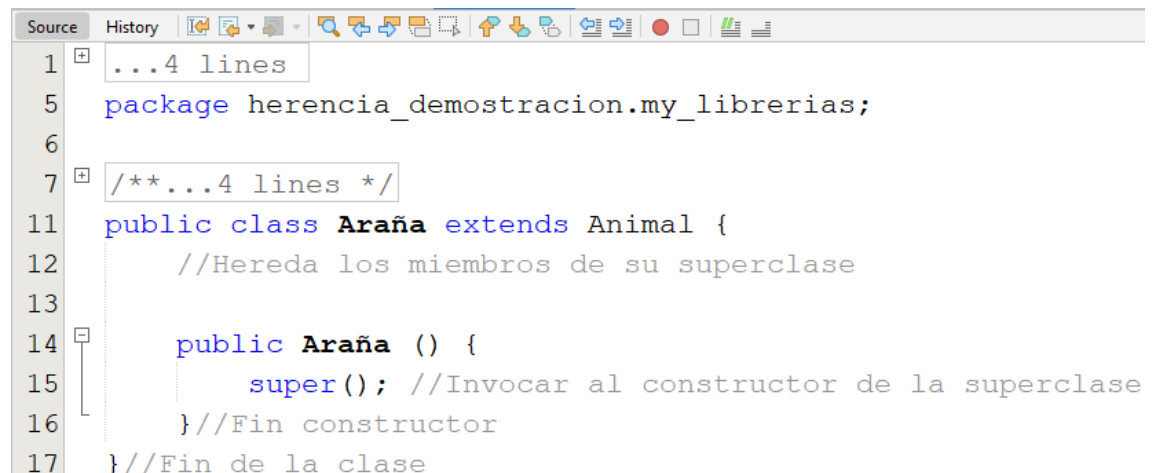
- Definiendo una clase base



```
1  ...4 lines
5  package herencia_demostracion.my_librerias;
6
7  /**...4 lines */
11 public class Animal {
12     // Variable de instancia (miembros, campos, atributos
13     protected int extremidades;
14     protected String raza;
15     protected String sexo; //Macho o Hembra
16
17     //Definición del constructor por default
18     public Animal () {
19         extremidades = 0;
20         raza = "ninguna";
21         sexo = "ninguno";
22     } //Fin método
23
24     public void caminar () {
25         System.out.println("Sin caminar");
26     } //Fin método
27
28     public void comer () {
29         System.out.println("Esperando alimento");
30     } //Fin método
31
32     public void dormir () {
33         System.out.println("En preparación para dormir");
34     } //Fin método
35 } //Fin de la clase
```

- Definiendo clases derivadas

Subclase denominada “**Araña**”, que hereda de la clase Animal



```
1  ...4 lines
5  package herencia_demostracion.my_librerias;
6
7  /**...4 lines */
11 public class Araña extends Animal {
12     //Hereda los miembros de su superclase
13
14     public Araña () {
15         super(); //Invocar al constructor de la superclase
16     } //Fin constructor
17 } //Fin de la clase
```

Subclase denominada “**Gato**” que hereda de la clase animal

```
Source History
1 ...4 lines
5 package herencia_demostracion.my_librerias;
6
7 /**...4 lines */
11 public class Gato {
12     //Hereda los miembros de la superclase
13     private String nombre;
14
15     public Gato () {
16         nombre = "kisifu";
17     } //Fin del constructor
18
19     public void jugar () {
20         System.out.println("Me gusta jugar con una bola de lana");
21     } //Fin de método
22
23 } //Fin de la clase
24
```

Subclase denominada “**Pescado**” que extiende de la clase Animal

```
Source History
1 ...4 lines
5 package herencia_demostracion.my_librerias;
6
7 /**...4 lines */
11 public class Pescado {
12     //Variable de instancia propia
13     private String tipo;
14
15     public Pescado () {
16         tipo = "pilapia";
17     } //Fin del constructor
18 } //Fin de la clase
19
```

### Realizo la solución al siguiente ejercicio aplicando la característica de herencia

Un ejemplo muy evidente en la disciplina de las matemáticas para la demostración de la herencia, es la parte de figuras geométricas.

Una figura geométrica parte de dos opciones, la primera que sea una figura de dos dimensiones y la segunda que sea una figura de tres dimensiones. En las figuras de dos dimensiones se pueden ubicar a los cuadrados, Triángulo y círculo, etc. En las figuras de tres dimensiones se pueden ubicar al, cubo, Tetraedro, etc. Por otra parte, las figuras Triángulos y Cuadrado se clasifican como polígonos. Los triángulos a su vez están clasificados en: Equilátero, Isósceles, Escaleno y Rectángulos.

El círculo a su vez, es parte de las figuras cónicas que puede extender de las Elipses, y recuerde que las Elipses en la geometría se ubican como figuras cónicas.

De acuerdo a lo planteado, considere la representación jerárquica de estas figuras y a partir de ello realice su implementación en el lenguaje de programación; diseñe los métodos para calcular su área y el perímetro, teniendo en cuenta sus detalles para aplicar la característica de herencia de la manera más correcta posible.

**Para la realización del ejercicio tome en cuenta lo siguiente:**

- La clasificación de cada una de las figuras geométricas.
- Las variables necesarias para cada una de las ecuaciones que conlleva al cálculo de su área y su perímetro.
- En el caso de las figuras geométricas tridimensionales recuerde que estas también poseen un volumen.
- Hacer la mayor reutilización de código que le sea posible para maximizar el uso de la herencia.
- Aplicar la sobreescripción de métodos, tomando en cuenta las funcionalidades necesarias que le permita calcular el área, el perímetro y el volumen en el caso de ser necesario.
- Recuerde el uso de los modificadores de acceso y las reglas de los constructores en las subclases.
- Para la clase triángulo complete las subclases necesarias de acuerdo a la clasificación de los triángulos.
- En la clase principal (clase main) o clase controladora, elabore un menú de opciones desde el cual se pueda mandar a ejecutar el programa solicitando el tipo de figura a la cual se le aplicará el cálculo del área y el perímetro, el volumen en el caso de las figuras que apliquen.
- Para un apoyo del ejercicio y la conformación de las figuras geométricas, visite el siguiente URL:

**<https://www.universoformulas.com/matematicas/geometria/figuras-geometricas/>**

