UNIVERSIDAD CÉSAR VALLEJO

PROGRAMACIÓN ORIENTADA A OBJETOS

FACULTAD DE INGENIERIA
ESCUELA PROFESIONAL DE INGENIERIA DE SISTEMAS

UNIVERSIDAD CÉSAR VALLEJO

PROGRAMACIÓN ORIENTADA A OBJETOS

Sesión 03

Herencia

Eric Gustavo Coronel Castillo
gcoronelc.blogspot.com

Herencia

Contenido :

- (T) Herencia. Definición y clasificación. Herencia desde una clase hacia sus instancias. Herencia desde una clase hacia sus sub-clases.
- (L) Creación de clases y subclases. Palabras reservadas: extends, this y super. Constructores en las subclases.

objetivos

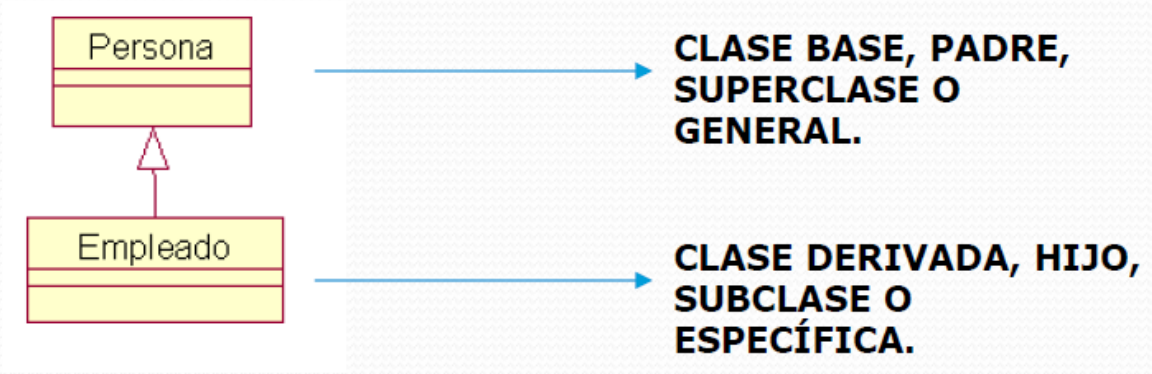
- Explicar los conceptos de herencia y su aplicación práctica en la solución de problemas.

HERENCIA

DEFINICIÓN

Herencia: Creación de clase a partir de clases existentes.

Es una de las características mas importantes de la programación orientada a objetos constituye un mecanismo que permite reutilizar atributos y métodos de Clases existentes. Fomenta la reutilización de software.



* El término herencia es utilizado en biología para referirse a ciertas características físicas y de comportamiento que son comunes entre familiares (padres e hijos).

HERENCIA

DEFINICIÓN

Herencia: Creación de clase a partir de clases existentes.

La herencia permite definir una clase (subclase) sobre la base de otra clase (superclase).

La subclase hereda aquellos miembros de la superclase (variables y métodos) que hayan sido declarados como public, protected o sin especificador de acceso.

Una superclase declara un miembro como protected para permitir el acceso al miembro desde el interior de sus subclases, a la vez que impide el acceso al miembro desde el exterior de la superclase.

El constructor no es un miembro y no es heredado por las subclases.

La nueva clase puede añadir nuevos miembros, incluso puede redefinir miembros de la superclase.

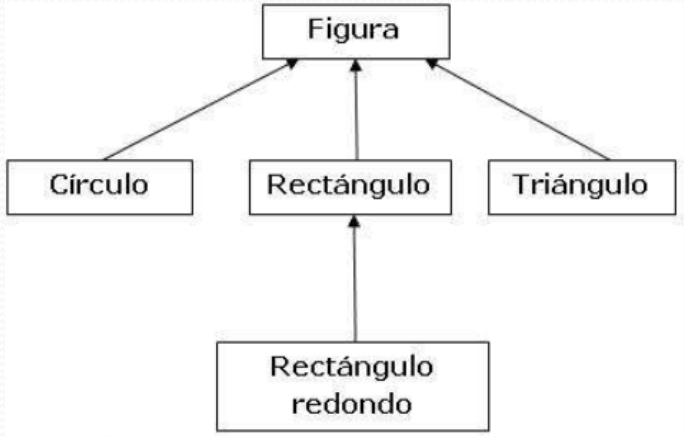
Herencia

- ## Clasificación de herencia

En java existen dos tipos de herencia, herencia simple y herencia múltiple.

Herencia simple
Un objeto puede extender las características de otro objeto y de ningún otro, es decir, que solo puede heredar o tomar atributos de un solo padre o de una sola clase.

Herencia múltiple
Un objeto puede extender las características de uno o más objetos, es decir, puede tener varios padres.



HERENCIA

APLICACIONES

La herencia permite que una clase (la subclase), tenga el comportamiento de otra clase (la superclase) y adicionalmente, si lo requiere, pueda modificar o agregar nuevos comportamientos y atributos.

1. *La subclase presenta un comportamiento adicional a su superclase (agrega atributos y/o métodos).*
2. *La subclase presenta el mismo comportamiento que su superclase.*
3. *La subclase presenta un comportamiento similar a su superclase (modifica métodos heredados y/o desactiva métodos heredados).*

En java, la jerarquía de Clases empieza con la Clase Object (ubicado en el paquete java.lang)

* La programación implementando el concepto de herencia es programación diferencial.

HERENCIA

IMPLEMENTACIÓN EN JAVA

Forma general de la declaración de una clase que hereda la funcionalidad de otra clase es :

```
class nombreDeSubclase extends nombreDeSuperclase{
        // Cuerpo de la clase
}
```

La sentencia reservada en Java para denotar que una clase era la funcionalidad de otra clase es la palabra *extends*.

Es importante indicar que el constructor de la superclase puede invocarse desde la subclase utilizando la palabra reservada *super*.

super (lista de argumentos);

Esta instrucción tiene que ser la primera sentencia a ejecutarse dentro del constructor de la subclase.

EJEMPLOS GENERALES

Un Mamífero es un Animal

Un Ave es un Animal

Una Vaca es un Mamífero

Un Pato es un Ave

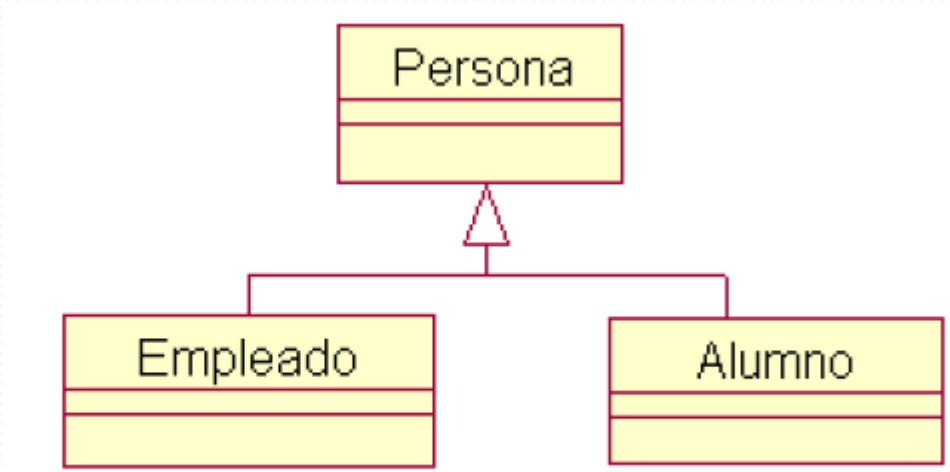
En Java, si una clase no tiene una superclase explícita, implícitamente su superclase será la clase **Object** de Java.

Lo que puede expresarse como:

```
class Animal { . . . }

class Mamifero extends Animal { . . . }

class Vaca extends Mamifero { . . . }

class Ave extends Animal { . . . }

class Pato extends Ave { . . . }
```

HERENCIA

EJEMPLO



EJEMPLO EN JAVA

APLICACIONES

1. La subclase presenta un comportamiento adicional a su superclase:

A continuación se muestran dos clases que implementan el concepto de herencia en el lenguaje de programación Java.

SUPERCLASE

```
public class Persona {
        protected String nombre;
        protected int edad;

        public Persona ( ) {
        }
        public int getEdad() {
           return edad;
        }
        public void setEdad(int edad) {
           this.edad = edad;
        }
        public String getNombre() {
           return nombre;
        }
        public void setNombre(String nom) {
           nombre = nom;
        }
}
```

SUBCLASE

```
public class Empleado extends Persona {
        private String RUC:

        public Empleado ( )  {
        }
        public String getRUC() {
           return RUC;
        }
        public void setRUC(String ruc) {
           RUC = ruc;
        }
}
```

EJEMPLO EN JAVA

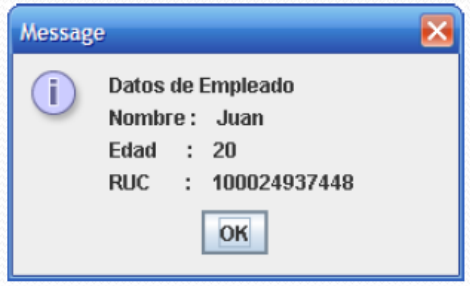
APLICACIONES

1. La subclase presenta un comportamiento adicional a su superclase:

A continuación se muestra una clase aplicación que permite crear un objeto de la clase Empleado creada anteriormente y se muestra como el objeto creado puede hacer uso de los métodos definidos en su clase y en su superclase.

CLASE APLICACIÓN

```
public class Prg1 {

   public static void main(String[] args) {

      Empleado objE=new Empleado();
      String salida="Datos de Empleado\n";

      objE.setNombre("Juan");
      objE.setEdad(20);
      objE.setRUC("100024937448");

      salida+="Nombre :   "+
objE.getNombre()+"\n";
      salida+="Edad     :   "+
objE.getEdad()+"\n";
      salida+="RUC      :   "+
objE.getRUC()+"\n";

JOptionPane.showMessageDialog(null,salida);

   }
}
```

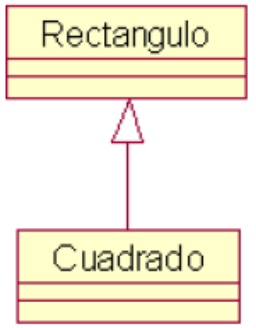


EJEMPLO EN JAVA

APLICACIONES

2. La subclase presenta el mismo comportamiento que su superclase:

Desarrolle la codificación en Java de la clase Rectángulo la cual tiene los atributos *base* y *altura*, un método constructor, métodos *setter* y *getter*, y un método llamado *area ( )* que calcula y devuelve el área del rectángulo.



SUPERCLASE

El acceso a los atributos de una superclase generalmente se establece en ***protected***.

SUBCLASE

```
public class Cuadrado extends Rectangulo {
        public Cuadrado (double lado) {
                super ( lado, lado ) ;
        }
}
```

* ***super*** permite referenciar el constructor de la clase base o superclase. En la clase Cuadrado la referencia ***super*** permite ejecutar el método constructor de la clase base o superclase Rectangulo.

Ejercicios Propuestos

1.- Se desea realizar una aplicación para ingresar los datos a empleados de una compañía, se sabe que se tienen 2 tipos de empleado a tiempo parcial y a tiempo completo.

Los datos del empleado de tiempo parcial son: Código, nombre, apellido, sueldo mes, RUC y horas trabajadas.

El sueldo se le asigna el número de horas trabajadas por una tarifa de 25 soles

Los datos del empleado a tiempo completo son: Código, nombre, apellido, sueldo mes, nombre de AFP y días trabajados.

El sueldo se le asigna por el número de días trabajados por una tarifa de 30 soles y un descuento del 12.5 % por AFP.


Datos de la clase padre
Código
Apellidos
Nombre
Empleado Tiempo Parcial
Empleado tiempo Completo
Empleado Tiempo Parcial
RUC
Horas Trabajadas
Calcular
Empleado Tiempo Completo
Nombre AFP
Numero CUPSS
Dias Trabajados

Ejercicios Propuestos

2.- Una empresa vende dos tipos de productos: Artefactos y Muebles a continuación muestran algunos de los productos que ofrece la empresa:

Artefactos:

Código	Descripción	Precio
100	Licuadora	170
101	Horno Microondas	350

Muebles:

Código	Descripción	Precio
200	Dormitorio	480
2001	Comedor	420

La empresa ofrece un % de descuento por producto solo para muebles. Si el monto a pagar es mayo a S/ 1000.00 el descuento es 10.75% caso contrario tendrá un descuento del 5%.

Diseñe un programa que registre 2 productos uno de tipo Artefacto y otro de tipo Muebles para lo cual debe de almacenar el código, cantidad y el % descuento. Además debe calcular el total a pagar por dichos productos.

Ejercicios Propuestos

3. -Se desea realizar una aplicación para ingresar los datos de los empleados de una clínica, se sabe que se tienen 2 tipos de empleado contratados y nombrados.

   Los datos del empleado contratado son: Código, nombre, apellido, especialidad, sueldo mes, RUC y horas trabajadas.

   El sueldo se le asigna el número de horas trabajadas por una tarifa de 25 soles.

   Los datos del empleado nombrado son: Código, nombre, apellido, especialidad, sueldo mes, nombre de AFP, N° CUPSS y días trabajados.

   El sueldo se le asigna por el número de días trabajados por una tarifa de 100 soles y un descuento del 10.5 % por AFP.

Ejercicios Propuestos

4.- Codificar un programa que defina la clase persona, que encapsule los datos nombre y DNI. Defínase la clase empleado, heredando de persona y que encapsule a su vez el salario. guardar los datos en vectores paralelos. Una vez guardadas 10 personas, mostrar en pantalla todos los datos.

Ejercicios Propuestos

5.- Utilizando como referencia la clase empleado del ejercicio anterior, créese una clase "oficina". Cada objeto oficina podrá tener un número indeterminado de empleados. La clase oficina, además, debe ofrecer un método que, según el tipo de trabajo del empleado, (a,b, ó c), ofrezca por pantalla un listado con la nómina (2 columnas: empleado y sueldo) teniendo en cuenta que a cada tipo de empleado le corresponden 1,000, 1,500 y 2,000 soles respectivamente. El programa debe pedir el número de empleados, instanciar entonces el objeto "oficina", pedir los datos de los empleados y visualizar la nómina.

Ejercicios Propuestos

6.- Una compañía de autobuses tiene tres tipos de autobús: el "larga distancia", el "interciudad" y el "metropolitano". A la vez, tiene cinco tipos de recorridos: urbanos, interurbanos, provinciales, nacionales e internacionales. Realizar un programa que lea por teclado una serie de datos en la forma: inicio, destino, kms totales; almacene estos datos asignándoles un código, y decida el tipo de recorrido y autobús y finalmente produzca un listado de los recorridos introducidos. Debe haber una jerarquía de clases para los autobuses y para los recorridos. Las funciones de impresión deben ser abstractas y estar declaradas en la clase base.

Ejercicios Propuestos

7.- Escribir un programa que acepte las medidas del lado de un rectángulo. Habrá tres clases en el sistema: rectbase, rectángulo y cuadrado. El programa deberá llamar al método estático en rectbase que se encargará, al pasarle las medidas, de instanciar un objeto adecuado. Una vez que la entrada termina, el programa debe listar los lados de cada instancia, y su área, cuyo método debe estar definido en rectbase.

Ejercicios Propuestos

8.- La empresa **NAUTILIUS S.A** tiene a su cargo la administración de 2 barcos**: Un crucero y un carguero.**

Por lo tanto se le pide a Ud. como programador, implementar un sistema que le permita al administrador de la empresa ingresar el nombre, código (de 5 dígitos), peso (en toneladas), y tamaño (en metros) de cada barco.

Finalmente, para cada tipo de barco, **a parte de lo ya mencionado**, se requiere lo siguiente:

Para el crucero: Se deberá registrar el nombre de los pasajeros que acudan a pagar su pasaje , sabiendo que pueden inscribirse como máximo **2500 personas**, se asignará a cada inscrito un código aleatorio de **4 dígitos** enteros validando que estos no se repitan y finalmente se ordenará por orden alfabético a todos los inscritos.

Para el carguero: Se deberá registrar el peso (en toneladas) de los contenedores a transportar, teniendo en cuenta que pueden ser como máximo 100 contenedores. Luego de ello se deberán agrupar los contenedores de 20 en 20 y eliminar el contenedor que tenga el mayor peso de cada grupo. Mostrar todos los pesos resultantes.

UNIVERSIDAD CÉSAR VALLEJO

Solo para los que quieren salir adelante