

GUÍA N° 3

LAB – 121

LABORATORIO DE INF – 121

“HERENCIA”

Gestión II/2016

Docente: Lic. Marcelo Aruquipa
Aux. Pamela Choque
Aux. Fabio Laura Yavi

PARTE I HERENCIA

La herencia es una propiedad de la P.O.O. que permite la definición, extensión de una clase a partir de otra ya existente, heredando la nueva clase todo de la clase ya existente.

Hay dos tipos de herencia:

- ✓ Herencia Simple, cuando la clase derivada se ha definido de una sola clase base.
- ✓ Herencia Múltiple, cuando una clase derivada se ha definido de más de una clase base.

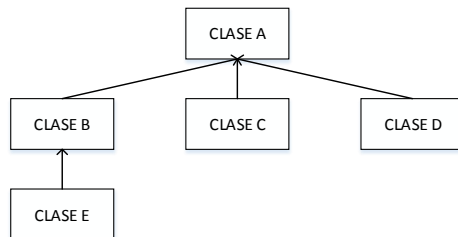
Obs.

Java y C# soportan la Herencia Simple, pero no así la Herencia Múltiple

En la Herencia se identifican dos tipos de clases:

- ✓ Clase Base (Superclase, Clase Padre), puede extender cualquier número de clases derivadas
- ✓ Clase Derivada (Subclases, Clase Hija), tiene una única clase base.

El concepto de herencia conduce a una estructura jerárquica de clases o estructura de árbol jerárquico, donde la Clase Base representa a un concepto general o amplio, en cambio la Clase Derivada representa a un concepto concreto, particular.



Donde:

- ✓ A es clase Base de B, C, D
- ✓ B es clase Base de E
- ✓ B, C y D clases Derivadas de A
- ✓ E clase Derivada de B

Sintaxis:

JAVA	C#
<pre> class NomBase{ ... public NomBase() { ... } } Clase NomDerivada extends NomBase{ ... public NomDerivada() { super () ; } } </pre>	<pre> class NomBase{ ... public NomBase() { ... } } Clase NomDerivada : NomBase{ ... public NomDerivada () : base () { ... } } </pre>

Uso del constructor de su clase base, se le denomina **super**

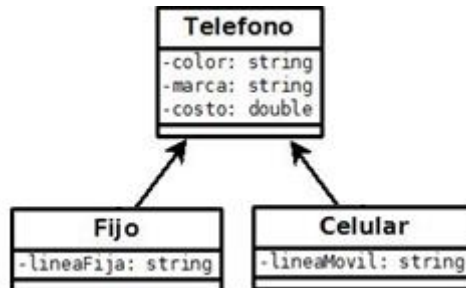
Uso del constructor de su clase base, se le denomina **base**

Obs.

- ✓ Se hereda **TODO** excepto constructores y destructor.
- ✓ En Java el constructor se conoce como **super**.
- ✓ En C# el constructor se conoce como **base**.
- ✓ **Base, super** se utilizan también para diferenciar métodos heredados que tengan el mismo nombre de los métodos de la clase de la clase derivada

1. EJEMPLO HERENCIA SIMPLE EN JAVA Y CSharpC#

Diagrama de clases



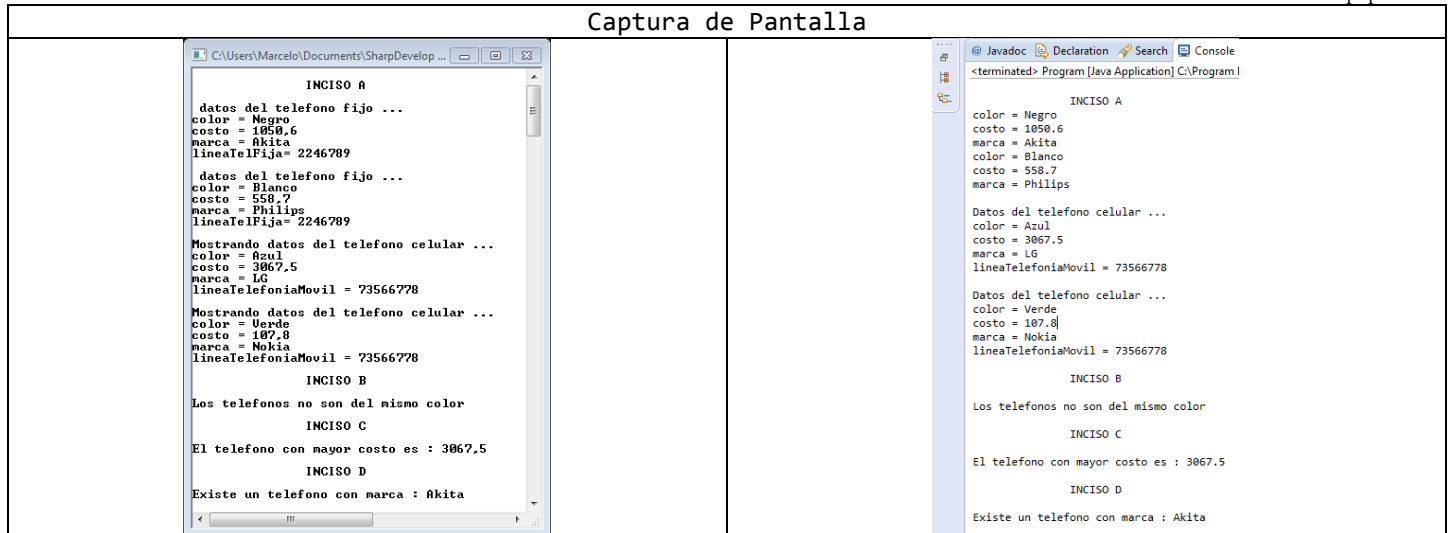
Dados 2 objetos de cada clase derivada:

- a. Crear constructores con argumentos para las clases derivadas
- b. Verificar cuales son del mismo color
- c. Mostrar cuál es el que tiene mayor costo.
- d. Verificar si existe algún teléfono de la marca z.

Telefono.cs	Telefono.java
<pre> using System; public class Telefono { protected string color; protected double costo; protected string marca; public Telefono(string c, double t, string m){ color = c; costo = t; marca = m; } public void Mostrar(){ Console.WriteLine("color = " + color); Console.WriteLine("costo = " + costo); Console.WriteLine("marca = " + marca); } public string Color { get { return color; } } public double Costo { get { return costo; } } public string Marca { get { return marca; } } } </pre>	<pre> public class Telefono { protected String color; protected double costo; protected String marca; public Telefono(String c, double t, String m){ color = c; costo = t; marca = m; } public void mostrar(){ System.out.println("color = " + color); System.out.println("costo = " + costo); System.out.println("marca = " + marca); } public String getColor() { return color; } public double getCosto() { return costo; } public String getMarca() { return marca; } } </pre>
Fijo.cs	Fijo.java
<pre> using System; public class Fijo : Telefono { private string lineaTelefoniaFija; //a public Fijo(string c, double t, string m, string l):base(c,t,m){ lineaTelefoniaFija = l; } public void Mostrar(){ Console.WriteLine("\n datos del telefono fijo ..."); base.mostrar(); Console.WriteLine("lineaTelFija= " + lineaTelefoniaFija); } } </pre>	<pre> public class Fijo extends Telefono { private String lineaTelefoniaFija; //a public Fijo(String c, double t, String m, String l){ super(c,t,m); lineaTelefoniaFija = l; } public void Mostrar(){ System.out.println("\n datos del telefono fijo ..."); super.mostrar(); System.out.println("lineaTelFija= " + lineaTelefoniaFija); } } </pre>

Celular.cs	Celular.java
<pre> using System; public class Celular:Telefono{ private string lineaTelefoniaMovil; //a public Celular(string c,double t,string m,string l):base(c,t,m){ lineaTelefoniaMovil = l; } public bool MismoColor(Telefono X){ return X.Color.Equals(color); } //b public void CompColor(Fijo A, Fijo B, Celular C){ if(MismoColor(A) && MismoColor(B) && MismoColor(C)) Console.WriteLine("Los telefonos son del mismo color"); else Console.WriteLine("Los telefonos no son del mismo color"); } //c public void MostrarMayorCosto(Fijo A, Fijo B, Celular C){ double maxi = Math.Max(costo, A.Costo); maxi = Math.Max(maxi, B.Costo); maxi = Math.Max(maxi, C.Costo); Console.WriteLine("El telefono con mayor costo es : "+maxi); } public bool Esmarca(Telefono Z, string x){ return Z.Marca.Equals(x); } //d public void VeriMarca(Fijo A, Fijo B, Celular C, string x){ if(Esmarca(A,x) Esmarca(B,x) Esmarca(C,x) Esmarca(this,x)) Console.WriteLine("Existe un telefono con marca : "+x); else Console.WriteLine("No existe un telefono con marca : "+x); } public void Mostrar(){ Console.WriteLine("\nDatos del telefono celular ..."); base.Mostrar(); Console.WriteLine("lineaTelefoniaMovil= "+lineaTelefoniaMovil); } } </pre>	<pre> public class Celular extends Telefono{ private String lineaTelefoniaMovil; //a public Celular(String c,double t,String m,String l){ super(c,t,m); lineaTelefoniaMovil = l; } public boolean mismoColor(Telefono X){ return X.getColor().equals(color); } //b public void compColor(Fijo A, Fijo B, Celular C){ if(mismoColor(A) && mismoColor(B) && mismoColor(C)) System.out.println("Los telefonos son del mismo color"); else System.out.println("Los telefonos no son del mismo color"); } //c public void mostrarMayorCosto(Fijo A, Fijo B, Celular C){ double maxi = Math.max(costo, A.getCosto()); maxi = Math.max(maxi, B.getCosto()); maxi = Math.max(maxi, C.getCosto()); System.out.println("El telefono con mayor costo es : "+maxi); } public boolean esMarca(Telefono Z, String x){ return Z.getMarca().equals(x); } //d public void veriMarca(Fijo A, Fijo B, Celular C, String x){ if(esMarca(A,x) esMarca(B,x) esMarca(C,x) esMarca(this,x)) System.out.println("Existe un telefono con marca : "+x); else System.out.println("No existe un telefono con marca : "+x); } public void mostrar(){ System.out.println("\nDatos del telefono celular ..."); super.mostrar(); System.out.println("lineaTelefoniaMovil = "+lineaTelefoniaMovil); } } </pre>
Program.cs	Program.java
<pre> using System; class Program{ public static void Main(string[] args){ //Inciso A Console.WriteLine("\n INCISO A"); Fijo A = new Fijo("Negro", 1050.6, "Akita", "2246789"); Fijo B = new Fijo("Blanco", 558.7, "Philips", "2246789"); Celular X = new Celular("Azul", 3067.5, "LG", "73566778"); Celular Y = new Celular("Verde", 107.8, "Nokia", "73566778"); A.Mostrar(); B.Mostrar(); X.Mostrar(); Y.Mostrar(); //Inciso B Console.WriteLine("\n INCISO B\n"); X.CompColor(A, B, Y); //Inciso C Console.WriteLine("\n INCISO C\n"); X.MostrarMayorCosto(A, B, Y); //Inciso D Console.WriteLine("\n INCISO D\n"); string z = "Akita"; X.VeriMarca(A, B, Y, z); Console.ReadKey(true); } } </pre>	<pre> public class Program { public static void main(String[] args) { //Inciso A System.out.println("\n INCISO A"); Fijo A = new Fijo("Negro", 1050.6, "Akita", "2246789"); Fijo B = new Fijo("Blanco", 558.7, "Philips", "2246789"); Celular X = new Celular("Azul", 3067.5, "LG", "73566778"); Celular Y = new Celular("Verde", 107.8, "Nokia", "73566778"); A.mostrar(); B.mostrar(); X.mostrar(); Y.mostrar(); //Inciso B System.out.println("\n INCISO B\n"); X.compColor(A, B, Y); //Inciso C System.out.println("\n INCISO C\n"); X.mostrarMayorCosto(A, B, Y); //Inciso D System.out.println("\n INCISO D\n"); String z = "Akita"; X.veriMarca(A, B, Y, z); } } </pre>

Captura de Pantalla

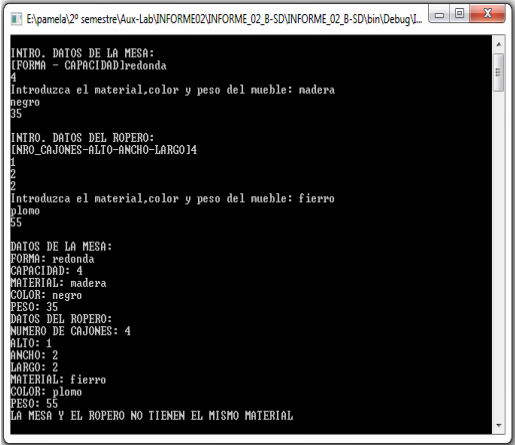
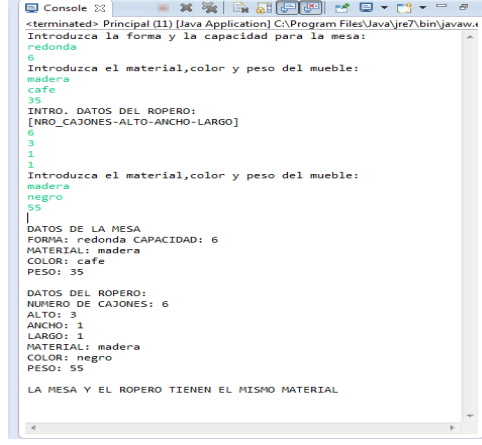


2. EJEMPLO

Sean las clases Ropero y Mesa, ambas heredan de la clase padre Mueble resolver los siguientes incisos:

- Crear el método leer para asignar los atributos a cada objeto.
- Crear el método mostrar para desplegar los datos de ambos muebles.
- Verificar si ambos tienen el mismo material.

Mueble.sc	Mueble.java	
<pre> public class Mueble { protected string material; protected string color; protected int peso; public void LeerM(){ Console.WriteLine("Intr. material,color, peso del mueble: "); this.material = Console.ReadLine(); this.color = Console.ReadLine(); this.peso=int.Parse(Console.ReadLine()); } public void MostrarM(){ Console.WriteLine("\nMATERIAL:"+ this.material); Console.WriteLine("\nCOLOR: "+this.color); Console.WriteLine("\nPESO: "+this.peso); } public string Material { get { return material; } } } </pre>	<pre> public class Mueble { protected String material; protected String color; protected int peso; public void leerM(){ Scanner lee = new Scanner(System.in); System.out.println("Intro. material,color, peso mueble: "); this.material = lee.next(); this.color = lee.next(); this.peso = lee.nextInt(); } public void mostrarM(){ System.out.println("MATERIAL:"+this.material); System.out.println("COLOR:"+this.color); System.out.println("PESO: "+this.peso); } public String getMaterial() { return material; } } </pre>	<pre> classDiagram class Mueble { - material : string - color : string - peso : int + Leer() + MostrarM() + getMaterial() } class Ropero { - NroCaj : int - Alto : int - Ancho : int - Largo : int + Leer() + Mostrar() } class Mesa { - Forma : string - Capacidad : int + Leer() + Mostra() + veriMaterial(Ropero r) } Mueble < -- Ropero Mueble < -- Mesa </pre>
Ropero.cs	Ropero.java	
<pre> public class Ropero : Mueble { private int nroCj; private int alto; private int ancho; private int largo; public void Leer(){ Console.WriteLine("\nINTRO. DATOS DEL ROPERO: "); Console.WriteLine("\n[NRO_CAJONES-ALTO-ANCHO-LARGO]"); this.nroCj=int.Parse(Console.ReadLine()); this.alto= int.Parse(Console.ReadLine()); this.ancho=int.Parse(Console.ReadLine()); this.largo=int.Parse(Console.ReadLine()); base.Leer(); } public void Mostrar() { Console.WriteLine("\nDATOS DEL ROPERO:"); } } </pre>	<pre> public class Ropero extends Mueble{ private int nroCj; private int alto; private int ancho; private int largo; public void leerR(){ Scanner lee = new Scanner(System.in); System.out.println("INTRO. DATOS DEL ROPERO: "); System.out.println("[NRO_CAJONES-ALTO-ANCHO-LARGO]"); this.nroCj = lee.nextInt(); this.alto = lee.nextInt(); this.ancho = lee.nextInt(); this.largo = lee.nextInt(); this.leer(); } public void mostrar() { System.out.println("\nDATOS DEL ROPERO:"); } } </pre>	

<pre> Console.WriteLine("\nNUMERO DE CAJONES: " +this.nroCj); Console.WriteLine("\nALTO: " +this.alto); Console.WriteLine("\nANCHO: " +this.ancho); Console.WriteLine("\nLARGO: " +this.largo); this.MostrarM(); } } </pre>	<pre> System.out.println("NUMERO DE CAJONES: " +this.nroCj); System.out.println("ALTO: " + this.alto); System.out.println("ANCHO " + this.ancho); System.out.println("LARGO: " + this.largo); this.mostrarM(); } } </pre>
<p style="text-align: center;">Mesa.cs</p> <pre> public class Mesa : Mueble { private string forma; private int capacidad; public void Leer() { Console.WriteLine("\nINTRO. DATOS DE LA MESA: "); Console.WriteLine("\n[FORMA - CAPACIDAD]"); this.forma = Console.ReadLine(); this.capacidad=int.Parse(Console.ReadLine()); this.LeerM(); } public void Mostrar() { Console.WriteLine("\nDATOS DE LA MESA:"); Console.WriteLine("\nFORMA: " +this.forma); Console.WriteLine("\nCAPACIDAD: " +this.capacidad); this.MostrarM(); } } public void VeriMaterial(Ropero r) { if(this.Material==(r.Material)) Console.WriteLine("\nMESA Y ROPERO MISMO MATERIAL"); else Console.WriteLine("\nMESA Y ROPERO DIFERENTE MATERIAL"); } } </pre>	<p style="text-align: center;">Mesa.java</p> <pre> public class Mesa extends Mueble{ private String forma; private int capacidad; public void leer(){ Scanner lee = new Scanner(System.in); System.out.println("Intro. Datos de la mesa"); System.out.println("Intro.forma - capacidad mesa"); this.forma = lee.next(); this.capacidad = lee.nextInt(); this.leer(); } public void mostrar() { System.out.println("\nDATOS DE LA MESA"); System.out.println("FORMA: " +this.forma); System.out.println("CAPACIDAD: " +this.capacidad); this.mostrar(); } } public void veriMaterial(Ropero r) { if(this.getMaterial().equals(r.getMaterial())) System.out.println("\nMESA Y ROPERO MISMO MATERIAL"); else System.out.println("\nMESA Y ROPERO DIFERENTE NO TIENEN EL MISMO MATERIAL"); } } </pre>
<p style="text-align: center;">Program.cs</p> <pre> public class Program { public static void Main() { Mesa m = new Mesa(); Ropero r = new Ropero(); m.Leer(); m.Mostrar(); r.Leer(); r.Mostrar(); m.VeriMaterial(r); Console.ReadKey(); } } </pre>	<p style="text-align: center;">Program.java</p> <pre> public class Program { public static void main(String[] args) { Mesa m = new Mesa(); Ropero r = new Ropero(); m.leer(); m.mostrar(); r.leer(); r.mostrar(); m.veriMaterial(r); } } </pre>
<p style="text-align: center;">Captura de Pantalla cs</p> 	<p style="text-align: center;">Captura de Pantalla java</p> 

EJERCICIOS.

- Dado n objetos Mesa, determinar cuantos tienen como peso x
- Dado n Roperos ordenar por el número de cajas