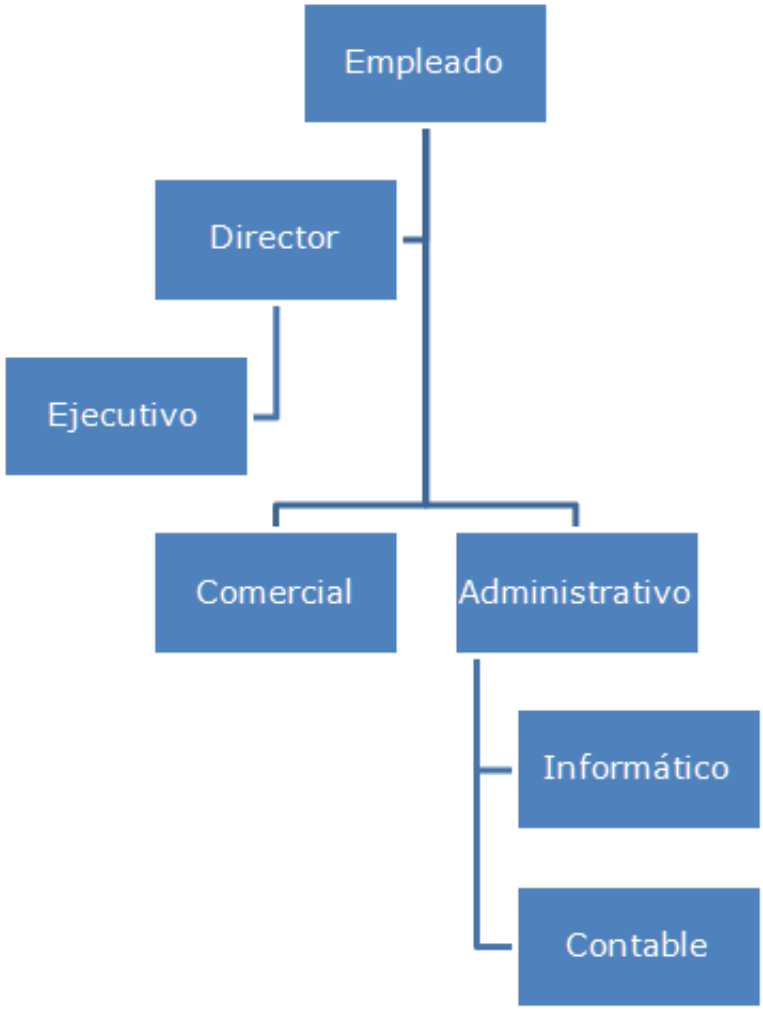
Herencia

La herencia es un mecanismo utilizado en la programación orientada a objetos que tiene por objeto extender la funcionalidad de una clase.
La clase superior (también llamada clase base o clase padre) es la clase de la cual se hereda. La clase hija es la clase que hereda de la clase superior. Se denomina también clase derivada.
Las clases hijas se diferencian de la clase padre agregando funcionalidad a ésta.

Se heredan todos los métodos y atributos de la clase base pero no los constructores. Sin embargo, cuando el compilador instancia una clase hija, debe instanciar también la clase padre.



A tener en cuenta

La herencia múltiple (heredar al mismo tiempo de más de una clase) no está permitida.

C#

```
class ClaseHija:ClasePadre
```

VB.NET

```
Public Class Hija
    Inherits Padre
```

Sobreescritura

La sobreescritura de métodos o atributos permite reemplazar la implementación de los miembros heredados a fin de extender y/o adaptar la funcionalidad específica de los mismos a las clases hijas (o clases derivadas).

En .NET es necesario autorizar la sobreescritura en los métodos o atributos que pueden ser sobreescritos en la clase padre y al mismo tiempo, declarar específicamente en la clse hija que se está sobrescribiendo determinado métdo o atributo.

C#

```
public virtual string MetodoPadreSobreescribible()

public override string MetodoPadreSobreescribible()
```

VB.NET

```
Public Overridable Function MetodoPadreSobreescribible() As String

Public Overrides Function MetodoPadreSobreescribible() As String
```

Ejecución de métodos de la clase base

Es posible, desde una clase hija, ejecutar las implementaciones originales de los métodos de la clase padre.

C#

```
public class Persona
{
    public Persona() { }

    public Persona(string paramNombre, string paramApellido)
    {
        this.Nombre = paramNombre;
        this.Apellido = paramApellido;
    }

    public string Nombre { get; set; }
    public string Apellido { get; set; }

    public virtual void MostrarValores()
    {
        Console.WriteLine("Nombre: {0}", Nombre);
        Console.WriteLine("Apellido: {0}", Apellido);
    }
}

public class Alumno: Persona
{
    public Alumno(string paramNombre, string paramApellido, string paramLegajo)
    {
        this.Nombre = paramNombre;
        this.Apellido = paramApellido;
        this.Legajo = paramLegajo;
    }

    public string Legajo { get; set; }

    public override void MostrarValores()
    {
        base.MostrarValores();
        Console.WriteLine("Legajo: {0}", Legajo);
    }
}
```

VB.NET

```
Public Class Persona

    Public Sub New()

    End Sub

    Public Sub New(ByVal paramNombre As String, ByVal paramApellido As String)

        Me.Nombre = paramNombre
        Me.Apellido = paramApellido

    End Sub

    Public Property Nombre As String
    Public Property Apellido As String

    Public Overridable Sub MostrarValores()

        Console.WriteLine("Nombre: {0}", Nombre)
        Console.WriteLine("Apellido: {0}", Apellido)

    End Sub

End Class

Public Class Alumno
    Inherits Persona

    Public Sub New(ByVal paramNombre As String,
                   ByVal paramApellido As String,
                   ByVal paramLegajo As String)

        Me.Nombre = paramNombre
        Me.Apellido = paramApellido
        Me.Legajo = paramLegajo

    End Sub

    Public Property Legajo As String

    Public Overrides Sub MostrarValores()

        MyBase.MostrarValores()
        Console.WriteLine("Legajo: {0}", Legajo)

    End Sub

End Class
```

Clases abstractas

Las clases abstractas son clases que no se pueden instanciar, aunque se puede heredar de ellas. Como su nombre sugiere, las clases abstractas se utilizan como base en una jerarquía de herencia, sirviendo como modelo que implementa una funcionalidad mínima de un objeto.
Los constructores de las clases abstractas deben declararse sin modificadores de visibilidad.

C#

```
public abstract class Vehiculo
{

    public Vehiculo()
    {
        Console.WriteLine("Se ha ejecutado el constructor de la clase abstracta.");
        Console.WriteLine();
    }

    public double Velocidad {get; set;}
    public int Capacidad { get; set; }

    public void Conducir()
    {
        Console.WriteLine("Se ha ejecutado el método Conducir de la clase abstracta.");
        Console.WriteLine();
    }
}
```

VB.NET

```
Public MustInherit Class Vehiculo

    Public Sub New()

        Console.WriteLine("Se ha ejecutado el constructor de la clase abstracta.")
        Console.WriteLine()

    End Sub

    Public Property Velocidad As Double
    Public Property Capacidad As Integer

    Public Sub Conducir()

        Console.WriteLine("Se ha ejecutado el método Conducir de la clase abstracta.")
        Console.WriteLine()

    End Sub

End Class
```

Miembros abstractos

Es posible declarar métodos o atributos abstractos en una clase abstracta. La finalidad es abstraerse de la implementación concreta de los mismos y delegar su implementación a las clases hijas.

C#

```
public abstract class Figura
{
    //Esta clase tiene un método abstracto.
    //El método abstracto sólo puede definirse, no implementarse.
    //Los métodos abstractos sólo pueden declararse en clases abstractas.

    public abstract double Superficie(double param);
}

class Circulo:Figura
{
    public override double Superficie(double paramRadio)
    {
        return Math.PI * Math.Pow(paramRadio, 2);
    }
}
```

VB.NET

```
Public MustInherit Class Figura

    'Esta clase tiene un método abstracto.
    'El método abstracto sólo puede definirse, no implementarse.
    'Los métodos abstractos sólo pueden declararse en clases abstractas.

    Public MustOverride Function Superficie(ByVal param As Double) As Double

End Class

Public Class Circulo
    Inherits Figura

    Public Overrides Function Superficie(ByVal paramRadio As Double) As Double

        Return Math.PI * Math.Pow(paramRadio, 2)

    End Function

End Class
```

Clases selladas

Las clases selladas con clases que impiden que otras hereden de ella. En otras palabras, no se puede extender su funcionalidad.

C#

```
public sealed class ItemFactura
{
    //Esta clase ha sido declarada sellada, con lo cual no se puede extender.
    public ItemFactura()
    {
        Console.WriteLine("Se ha ejecutado el constructor de clase ItemFactura.");
        Console.WriteLine();
    }
}
```

VB.NET

```
Public NotInheritable Class ItemFactura

    'Esta clase ha sido declarada sellada, con lo cual no se puede extender.

    Public Sub New()

        Console.WriteLine("Se ha ejecutado el constructor de clase ItemFactura.")
        Console.WriteLine()

    End Sub

End Class
```
