**HERENCIA**

La programación orientada a objetos nos permite traducir el mundo que nos rodea a código. Esto incluye objetos físicos de la vida real y también objetos conceptuales.

Ahora bien, la Herencia representa el tercer pilar de la programación orientada a objetos (POO de ahora en adelante).

Los cuatro pilares de la POO son:  
-Abstracción (La capacidad de declarar clases)  
-Encapsulamiento (La capacidad de utilizar modificadores de acceso)  
-Reutilización (La capacidad de generar Herencia)  
-Polimorfismo (es la capacidad de permitir mediante la herencia que dos objetos distintos se comporten como un objeto padre común, por ejemplo en el uso de sobrecargas de métodos)

Entonces, la Herencia nos permite generalizar características y comportamientos en grupos más amplios. De manera que podamos reutilizar en la mayor cantidad de características comunes.

Una estrategia para poder identificar la Herencia, es por ejemplo analizando nuestras entidades para detectar redundancia de dichas características.

Para programar orientado a objetos es necesario primero desarrollar capacidad de análisis orientado a objetos. Para esto se lleva un paso más allá el principio de divide y vencerás de la metodología de resolución de problemas de la programación estructurada; de manera que podemos analizar de la manera más atómica los objetos que se encuentran involucrados en el sistema que queremos diseñar.

Lo más recomendable antes de empezar a programar es primero diseñar la arquitectura de nuestra aplicación mediante diagramas. Para detectar la herencia podemos usar por ejemplo diagramas de clases de UML. Ver la arquitectura de nuestras entidades nos va a permitir corregir errores de lógica antes de comenzar a programar y entender cómo van a componerse y relacionarse mediante herencia nuestros tipos definidos por el usuario (clases).

**Ejemplo**

public class Humano{

   // Aqui puedes agregar atributos y métodos

}

public class Arquero extends *Humano*{

   // Aquí la clase Arquero hereda atributos y métodos de su superclase Humano

   // Aunque también puedes agregar más atributos y métodos para esta clase

}