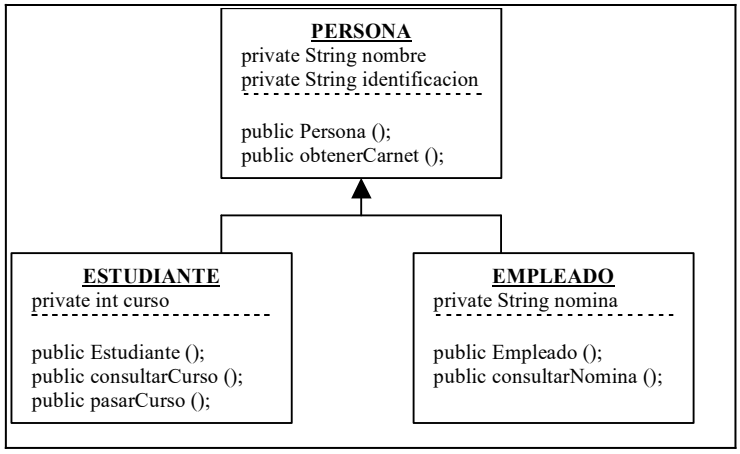
Uno de los objetivos fundamentales de la programación orientada a objetos es la de facilitar la ***reutilización del código***.

Ello permite volver a utilizar elementos que al haber sido ya realizados son bien conocidos y están, posiblemente, probados de forma exhaustiva.

La reutilización de código ha sido uno de los objetivos buscados con la creación de los nuevos lenguajes de programación. En particular, en los lenguajes de programación orientados a objetos el **mecanismo básico para la reutilización de código es la *herencia***.

A continuación podemos observar diversas definiciones de herencia:

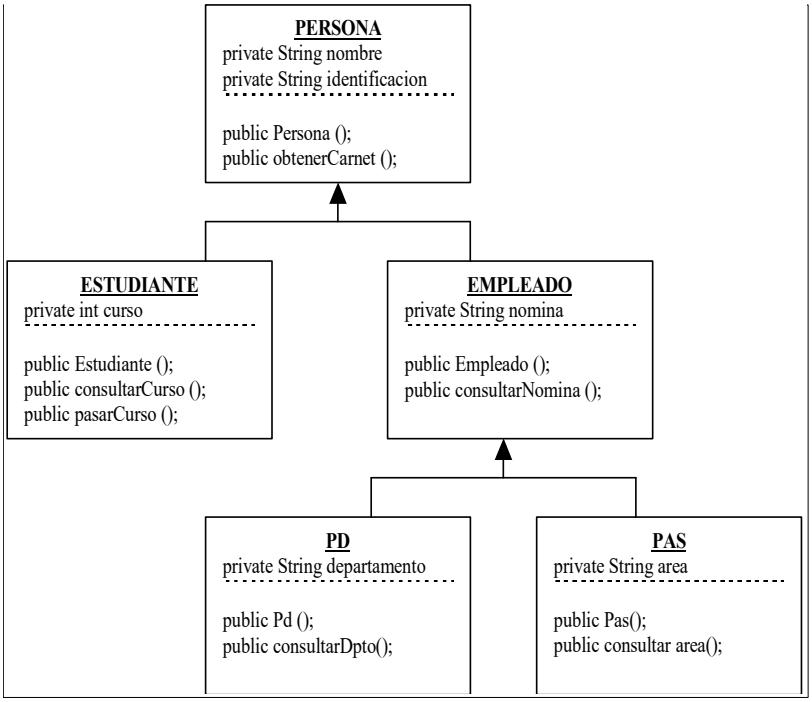
- Es el mecanismo por el cual elementos más específicos incorporan la estructura y el comportamiento definido en elementos más generales.
- La herencia es la técnica que permite definir una clase como especialización de otras ya definidas.
- Es el mecanismo mediante el cual se definen nuevos elementos como **extensión** o especialización de otros ya definidos.



En ocasiones, necesitamos una relación entre clases del tipo "es un". Por ejemplo, si quisiéramos modelar el **entorno de un instituto**, podíamos definir el objeto ***persona***, del que conocemos los atributos nombre e identificación, para representar a las personas que pertenecen al instituto. Pero también podemos "concretar" en la definición de *persona*, diferenciando aquellos que son ***estudiantes*** por un lado y ***empleados*** por otro, que podrían representarse también mediante clases.

La relación establecida crea una jerarquía llamada *estructura de clases* o *jerarquía de especialización/generalización.* En el nivel más alto de la jerarquía tenemos **la superclase (PERSONA)**, y descendiendo en la jerarquía vamos obteniendo **especializaciones de la superclase**, las subclases **(ESTUDIANTE y EMPLEADO)**, o clases descendientes de la superclase.

A su vez, **las subclases pueden asumir el papel de superclase**, si poseen clases descendientes en la estructura de clases. Éste sería el caso de la clase empleado si diferenciáramos entre ***personal docente*** *(PD)* ***y servicios*** *(PAS)*. A continuación se representa el nuevo nivel de la jerarquía:



Define las clases necesarias para crear objetos Alumnos y profesores.

La clase derivada o subclase hereda todos lo atributos y métodos de la superclase. De esta forma se dice que la nueva clase ***extiende*** a la clase original.

La relación entre los elementos de una subclase y los de la superclase se pueden resumir de la forma siguiente:

- ✓ Todo atributo o método de la superclase que no aparezca en la definición de la subclase es heredado sin modificación alguna, exceptuando **los constructores**, ya que en **cada nueva clase, incluso las derivadas, deben definir los suyos propios**. Si no se implementa ningún constructor, entonces se genera uno sin argumentos por defecto.
- ✓ **Los métodos** de la superclase que se definan de nuevo **en la clase derivada se sobrescriben**. Esta definición se aplicará a los objetos de la clase derivada.
- ✓ En la clase derivada pueden añadirse atributos (que generalmente serán privados) y métodos adicionales.

Crear las clases, metodos, atributos necesarios para crear la estructura anterior.