3.2 Herencia

* Una clase hereda a otra clase.
* Por principio de jerarquía la clase más alta del árbol se llama SUPERCLASE, CLASE BASE o CLASE PADRE.
* Las clases después del nivel 0 son SUBCLASES, CLASES DERIVADAS o CLASES HIJAS.
* Las clases superiores son clases genéricas y mientras que más abajo del árbol, las clases se vuelven más especializadas.
* La notación UML de herencia es una flecha con línea continua

Indicador de Herencia

* Para saber si se puede utilizar herencia hay que contestar afirmativamente la pregunta:

Clase A

Clase B

La clase B es un tipo

de clase A?

Si la respuesta es negativa entonces utiliza agregación.

* En java la SUPERCLASE = object.
* En java la palabra reservada para herencia es *extends.*

3.2.1 Herencia Simple

Un padre y uno o más hijos.

Object

Personaaaa

Alumno

Conclusiones

1. El constructor del hijo llama al de su padre. Entonces los constructores se mandan a llamar de abajo hacia arriba en la jerarquía con la palabra reservada *super(args).*
2. El método toString muestra el estado de toda la jerarquía.
3. toString se sobrescribe en toda la jerarquía.
4. equals compara todo el estado de la jerarquía debe sobrescribirse en toda ella.
5. Los objetos se crean de arriba hacia abajo (se ejecutan los constructores).
6. Primero existe la SuperClase (Object).
7. Se destruye de abajo hacia arriba en la jerarquía.
8. Los métodos heredados se usan sin restricción (sin instancia).
9. Se puede sobrecargar métodos en la jerarquía.
10. Si se duplica un método en la(s) clase(s) hija(s) se está sobrescribiendo.
11. Si un método es público se ve en la jerarquía y afuera, si es protegido solo se ve en la jerarquía y si es privado solo se puede ver en la clase.