Ejemplo A

|  |
| --- |
|  |

Ejemplo B

|  |
| --- |
|  |

**Fuente: Horstmann, C. S. (2016). *Core Java, Volume I – Fundamentals.***

* En ocasiones, necesitamos convertir tipos primitivos a un tipo objeto, como, por ejemplo: un tipo int a Integer.
* Todos los datos primitivos tienen Clases que representan su contraparte.
* Por ejemplo, la clase Integer corresponde al tipo de dato primitivo int.
* Este tipo de clases son usualmente llamadas ***wrappers*** (envoltorias).
* Las clases ***wrappers*** son: **Integer, Long, Float, Double, Short, Byte, Character y Boolean**
* Las primeras seis heredan una súper clase en común: Number.
* Las clases ***wrapper*** son inmutables, you cannot change a wrapped value after the wrapper has been constructed.
* También son clases finales, no podemos crear sub clases a partir de ellas.
* Supongamos que necesitamos un array list de enteros.
* Desafortunadamente, el tipo de dato dentro de los símbolos <> no puede ser un tipo de dato primitivo.
* No es posible crear un ***ArrayList<int>***
* Aquí, la clase wrapper Integer entra en acción. Es correcto crear un array list de objetos Integer.
* ***ArrayList<Integer> lista = new ArrayList<>()***

|  |
| --- |
|  |

* Afortunadamente, existe una característica muy útil nos facilita agregar un elemento de tipo ***int*** a nuestra lista ***ArrayList<Integer>***
* *lista.add(3)*
* La línea anterior es automáticamente trasladada a;
* *lista.add( Integer.valueOf(3) )*
* Esta conversión es llamada ***autoboxing.***

|  |
| --- |
|  |

* De manera inversa, cuando asignamos un objeto Integer a un valor int, el ***unboxing*** es automáticamente realizado.
* int n = lista.get(i)
* su equivalente, si no se realizara el ***unboxing*** automáticamente, sería:
* int n = lista.get(i).intValue();