# Ejecución 7 del 6 de mayo

# Tipos de datos en java

Java cuenta con un pequeño conjunto de tipos de datosprimitivos. Podríamos considerarlos fundamentales, ya que la mayor parte de los demás tipos, los tipos estructurados o complejos, son composiciones a partir de estos más básicos. Estos tipos de datos primitivos sirven para gestionar los tipos de información más básicos, como números de diversas clases o datos de tipo verdadero/falso (también conocidos como "valores booleanos" o simplemente "booleanos").

De estos tipos primitivos, ocho en total, seis de ellos están destinados a facilitar el trabajo con números. Podemos agruparlos en doscategorías: tipos numéricos enteros y tipos numéricos en punto flotante. Los primeros permiten operar exclusivamente con números enteros, sin parte decimal, mientras que el segundo grupo contempla también números racionales o con parte decimal.

**byte:** Representa un tipo de dato de 8 bits con signo. De tal manera que puede almacenar los valores numéricos de -128 a 127 (ambos inclusive).

**short:** Representa un tipo de dato de 16 bits con signo. De esta manera almacena valores numéricos de -32.768 a 32.767.

**int:** Es un tipo de dato de 32 bits con signo para almacenar valores numéricos. Cuyo valor mínimo es -231 y el valor máximo 231-1.

**long:** Es un tipo de dato de 64 bits con signo que almacena valores numéricos entre -263 a 263-1

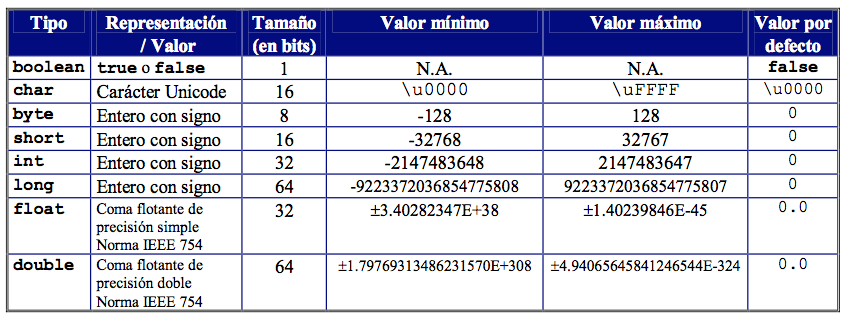
**float:** Es un tipo dato para almacenar números en coma flotante con precisión simple de 32 bits.

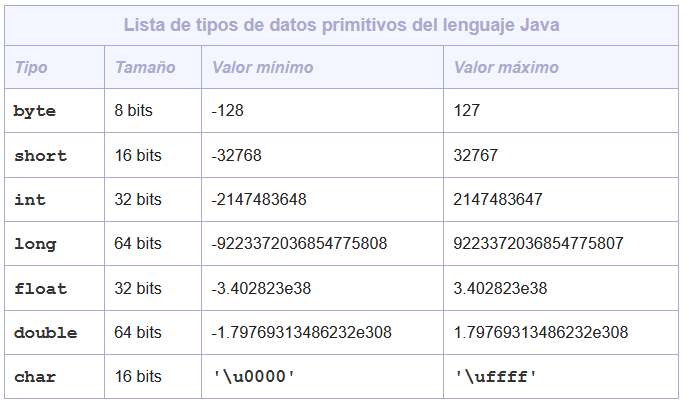
**double:** Es un tipo de dato para almacenar números en coma flotante con doble precisión de 64 bits.

**boolean:** Sirve para definir tipos de datos booleanos. Es decir, aquellos que tienen un valor de true o false. Ocupa 1 bit de información.

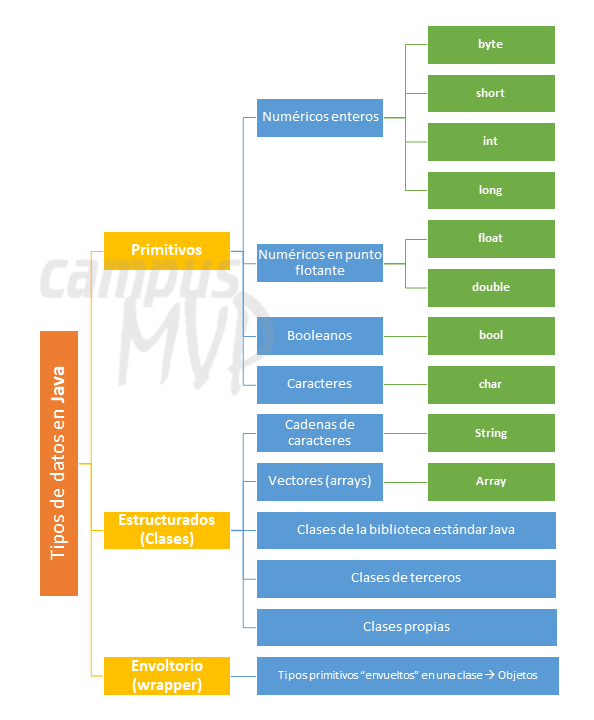
**char:** Es un tipo de datos que representa a un carácter Unicode sencillo de 16 bits.

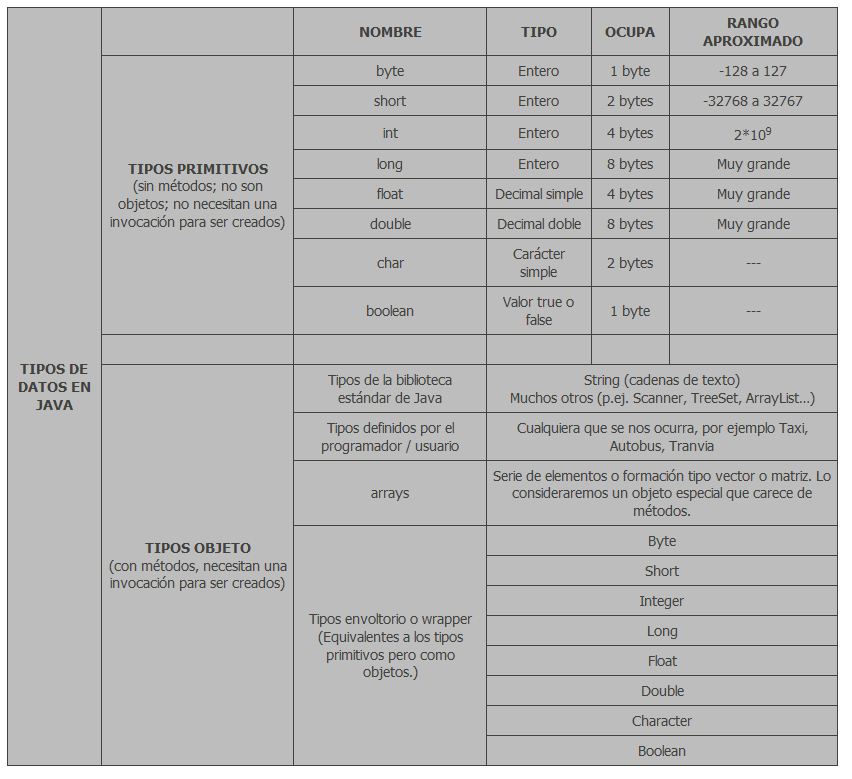
# Tamaño y rango de valores





# Grafico donde se engloban los tipos de datos, desde el más grande a los más pequeños



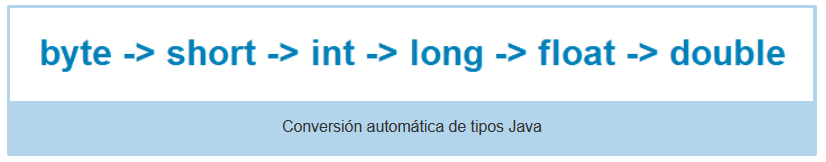


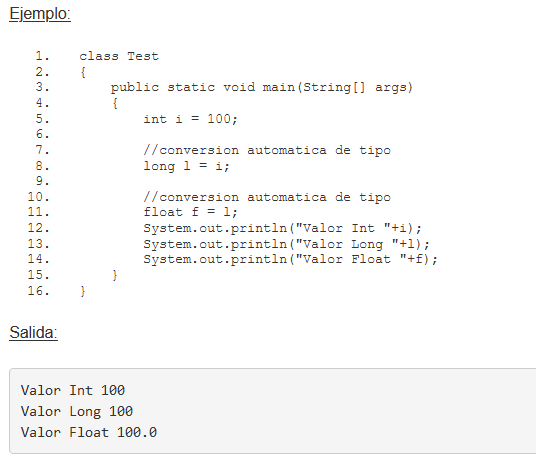
# Posibilidad de conversión de tipos de datos

### 1. Conversión automática de tipos o ampliación

La conversión de ampliación tiene lugar cuando dos tipos de datos se convierten automáticamente. Esto sucede cuando:

* Los dos tipos de datos son compatibles.
* Cuando asignamos el valor de un tipo de datos más pequeño a un tipo de datos más grande.

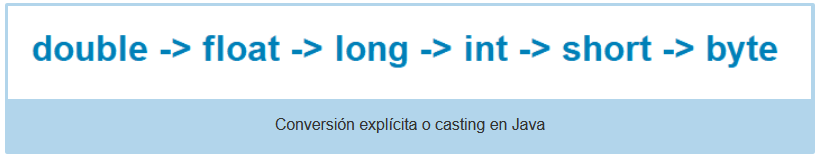


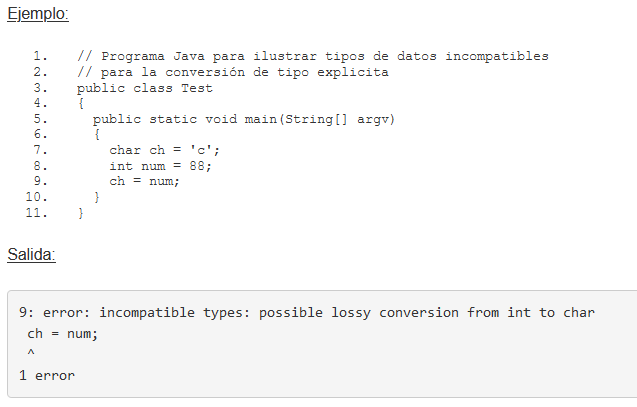


### 2. Conversión explícita de tipo o casting

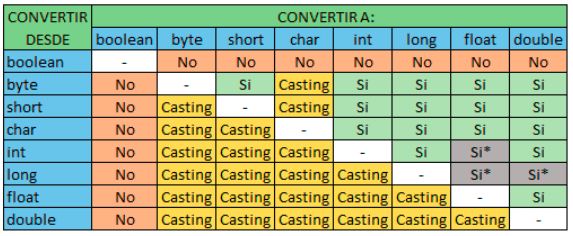
Si queremos asignar un valor de tipo de dato más grande a un tipo de dato más pequeño, realizamos un casteo/castingo lo que se conoce comoconversión de tipo explícito.

* Esto es útil para tipos de datos incompatibles donde la conversión automática no se puede realizar.
* char y int no son compatibles entre sí. Veamos luego cuando tratamos de convertir uno en otro.





### 3. Tabla de compatibilidad y conversión de datos



# Hacer una aplicación que pida 3 números al usuario, además deberá ordenarlos de mayor a menor y de menor a mayor. (Se adjunta proyecto en carpeta)

**package** clase;

**import** java.util.Scanner;

**public** **class** inicio {

**public** **static** **void** main(String[] args) {

Scanner lector = **new** Scanner(System.***in***);

System.***out***.println("Ingrese el primer número: ");

**int** variable1 = lector.nextInt();

System.***out***.println("Ingrese el segundo número: ");

**int** variable2 = lector.nextInt();

System.***out***.println("Ingrese el tercer número: ");

**int** variable3 = lector.nextInt();

**if** (variable1 > variable2 && variable2 >variable3) {

System.***out***.println("De mayor a menor: "+variable1+", "+variable2+", "+variable3);

System.***out***.println("De menor a mayor: "+variable3+", "+variable2+", "+variable1);

} **else** **if** (variable1 > variable3 && variable3 > variable2) {

System.***out***.println("De mayor a menor: "+variable1+", "+variable3+", "+variable2);

System.***out***.println("De menor a mayor: "+variable2+", "+variable3+", "+variable1);

} **else** **if** (variable2 > variable1 && variable1 > variable3) {

System.***out***.println("De mayor a menor: "+variable2+", "+variable1+", "+variable3);

System.***out***.println("De menor a mayor: "+variable3+", "+variable1+", "+variable2);

} **else** **if** (variable2 > variable3 && variable3 > variable1) {

System.***out***.println("De mayor a menor: "+variable2+", "+variable3+", "+variable1);

System.***out***.println("De menor a mayor: "+variable1+", "+variable3+", "+variable2);

} **else** **if** (variable3 > variable1 && variable1 > variable2) {

System.***out***.println("De mayor a menor: "+variable3+", "+variable1+", "+variable2);

System.***out***.println("De menor a mayor: "+variable2+", "+variable1+", "+variable3);

} **else** **if** (variable3 > variable2 && variable2 > variable1) {

System.***out***.println("De mayor a menor: "+variable3+", "+variable2+", "+variable1);

System.***out***.println("De menor a mayor: "+variable1+", "+variable2+", "+variable3);

}

lector.close();

}