1 | Tipos primitivos

En el momento en que se elige utilizar un sistema físico para realizar cómputos, éstos quedan sujetos a las leyes que rigen estos sistemas. Al elegir utilizar sistemas binarios con dos estados distinguibles, dichos cómputos quedan encuadrados por la lógica booleana; más aún, por una lógica booleana con un número grande, pero finito de símbolos disponibles.

Varios lenguajes de programación nos permiten acceder en forma privilegiada a aquellos tipos de datos que son representables con mayor facilidad dentro de este encuadre y gozan de una implementación particularmente eficiente, a estos tipos se les conoce como *tipos primitivos* y son los átomos a partir de los cuales se manejará cualquier tipo de información.

Meta

Que el alumno se familiarice con la representación en la computadora de los tipos primitivos de Java.

Objetivos

Al finalizar la práctica el alumno será capaz de:

- Utilizar apropiadamente el *casting* de tipos primitivos.
- Preveer los efectos de convertir datos de un tipo a otro tipo.
- Empacar y desempacar bits de información dentro de un tipo con más bits.

Antecedentes

Java tiene ocho tipos primitivos, los cuales se muestran en la Tabla 1.1.

Tabla 1.1: Tipos básicos de Java

Nombre	Tamaño	Representación en memoria
`boolean`	2 bytes	Booleano `true` o `false`
`char`	2 bytes	Caracter Unicode 2.0
`byte`	1 byte	Entero con signo en complemento a 2
`short`	2 bytes	Entero con signo en complemento a 2
`int`	4 bytes	Entero con signo en complemento a 2
`long`	8 bytes	Entero con signo en complemento a 2
`float`	4 bytes	Racional de acuerdo al estándar IEEE 754-1985
`double`	8 bytes	Racional de acuerdo al estándar IEEE 754-1985

* Un byte son 8 bits

EJERCICIOS

1. Junto con esta práctica se te entrega el código de la clase `ImpresoraBinario`. Utiliza para imprimir en pantalla la representación con bits de los siguientes números:

   - El `int` 456
   - El mismo número pero en una variable tipo `long`
   - El mismo número pero en una variable tipo `float`
   - El mismo número pero en una variable tipo `double`

   Explica ¿qué diferencias observas?

2. Repite los mismos pasos, pero ahora con -456

3. Repite los mismos pasos, pero con -456.601. Ojo, en este caso deberás hacer un casting para guardar el número en los tipos correspondientes a enteros y perderás la parte fraccionaria. ¿Qué número queda almacenado?

4. Realiza ahora pruebas con los operadores de corrimiento `«, », »>`. Recorre los números del iniciso anterior por uno y tres bits. Se usa así `int resultado = num « 3;`, imprime tus resultados. Asegúrate de que entiendes lo que hizo cada operador.

5. Finalmente, crea un `int` llamado `máscara` cuyos últimos cuatro dígitos sean unos. Ahora utiliza el operador de corrimiento necesario, para colocarlos en las posiciones 4 a la 7 (contado de derecha a izquierda). ¿Qué número obtienes?

6. Dado un `int num = 1408`, realiza la operación `num & máscara`. Imprime los bits resultantes y el valor numérico de tu resultado. Repite el ejercicio con `|` y `~`.