# Apunte actividad 4

## Expresiones vs. Instrucciones

Es fundamental entender la diferencia entre estos dos conceptos:

- **Expresión**: Es cualquier combinación de valores, variables, operadores y métodos que se evalúa para producir un único valor. Una expresión por sí sola no realiza una acción, solo calcula un resultado.
    - **Ejemplos**:
        - 5 + 3 (se evalúa a 8)
        - nombre + " " + apellido (se evalúa a un String completo)
        - edad > 18 (se evalúa a true o false)
- **Instrucción**: Es una línea de código que realiza una acción. Una instrucción termina con un punto y coma `;`. A menudo, una instrucción puede contener una o más expresiones.
    - **Ejemplos**:
        - int numero = 10; (declara e inicializa una variable)
        - System.out.println("Hola"); (imprime un valor en la consola)
        - if (edad > 18) { ... } (una estructura de control)

## Pruebas de Escritorio

Una **prueba de escritorio** es una técnica manual para simular la ejecución de un algoritmo o un fragmento de código, línea por línea. Consiste en crear una tabla donde anotas el valor de cada variable en cada paso. Esta técnica es invaluable para:

- **Entender la lógica** de un algoritmo.
- **Encontrar errores** o "bugs" de lógica antes de ejecutar el código.
- **Predecir el resultado** del programa con diferentes entradas.

**Cómo hacer una prueba de escritorio:**

1. Dibuja una tabla con una columna para el número de línea de código y una columna para cada variable.
2. Ve ejecutando el código mentalmente, línea por línea.
3. En cada línea, actualiza los valores de las variables en la tabla.

4. Anota también cualquier salida que se imprima en la pantalla.

## Constantes

Una **constante** es una variable cuyo valor no puede ser modificado una vez que ha sido inicializado. En Java, se declaran con la palabra clave **final**. Usar constantes es una buena práctica porque:

- **Mejora la legibilidad**: Evitas los "valores mágicos", que son números o cadenas de texto que aparecen en el código sin ninguna explicación. En su lugar, usas un nombre descriptivo (ej. PI).
- **Facilita el mantenimiento**: Si necesitas cambiar un valor que se usa en varias partes del programa (ej. un impuesto), solo tienes que modificar la constante en un único lugar.

**Ejemplo:**

```
final double PI = 3.14159;
final int DIAS_SEMANA = 7;
final String BIENVENIDA = "¡Hola, bienvenido!";

// No puedes hacer esto:
// PI = 3.14; // Esto causaría un error de compilación
```

## Casting

El **casting** es el proceso de convertir un valor de un tipo de dato a otro. Existen dos tipos principales:

1. **Casting Implícito (Automático)**: Ocurre cuando conviertes de un tipo de dato "pequeño" a uno "grande" sin pérdida de información (ej. de int a double). El compilador lo hace automáticamente.

```
int numeroEntero = 10;
double numeroDecimal = numeroEntero; // Casting automático, numeroDecimal es 10.0
```

2. **Casting Explícito**: Ocurre cuando conviertes de un tipo de dato "grande" a uno "pequeño", lo que podría implicar una pérdida de información (ej. de double a int). Debes forzar la conversión usando paréntesis con el tipo de destino (tipo).

```
double numeroDecimal = 9.99;
int numeroEntero = (int) numeroDecimal; // Casting explícito, numeroEntero es 9.
```

**Cuidado con la División**: En Java, si divides dos números enteros, el resultado será un entero. Para obtener decimales, debes asegurarte de que al menos uno de los números sea un tipo de dato de punto flotante (double o float), lo que se logra con un casting.

**Ejemplo:**

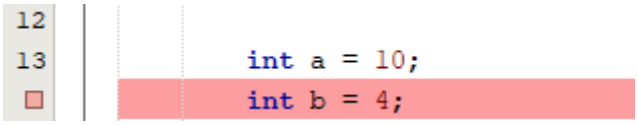
```java
int a = 10;
int b = 4;

double resultadoIncorrecto = a / b;          // Resultado es 2.0 (se pierden los decimales)
double resultadoCorrecto = (double) a / b;   // Resultado es 2.5
```

## Debugging

El **debugging** es el proceso de encontrar y corregir errores en tu código. En lugar de ejecutar el programa de golpe, un depurador (debugger) te permite ejecutar el código línea por línea, inspeccionar el valor de las variables y ver el flujo de ejecución.

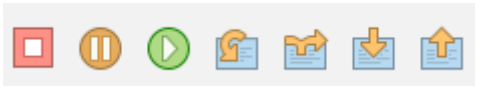
**Pasos para hacer debugging en NetBeans:**

1. **Punto de interrupción (Breakpoint)**: Coloca un punto de interrupción haciendo clic en el margen izquierdo de la línea de código donde quieres que el programa se detenga.

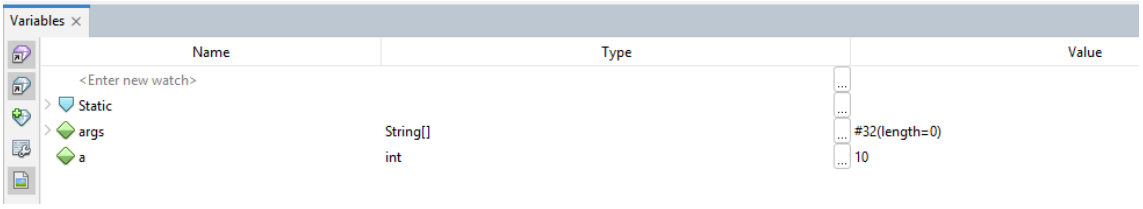



2. **Iniciar el Debugger**: En lugar de hacer clic en "Run", haz clic en el botón "Debug".

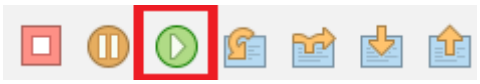
3. **Ejecutar paso a paso**: Utiliza los botones de control (como Step Over, Step Into) para avanzar una línea a la vez.



4. **Inspeccionar variables**: La ventana de "Variables" de NetBeans mostrará el valor actual de cada variable en el punto donde se detuvo el programa.




5. **Continuar**: Cuando hayas inspeccionado lo suficiente, puedes continuar la ejecución normal del programa.



El debugging es una de las herramientas más poderosas para cualquier programador. ¡No tengas miedo de usarla!