Jose Ignacio Zaballos

Programación Orientada a Objetos en Java (Parte I)

TFL – Tecnologías de la Información y la Comunicación II

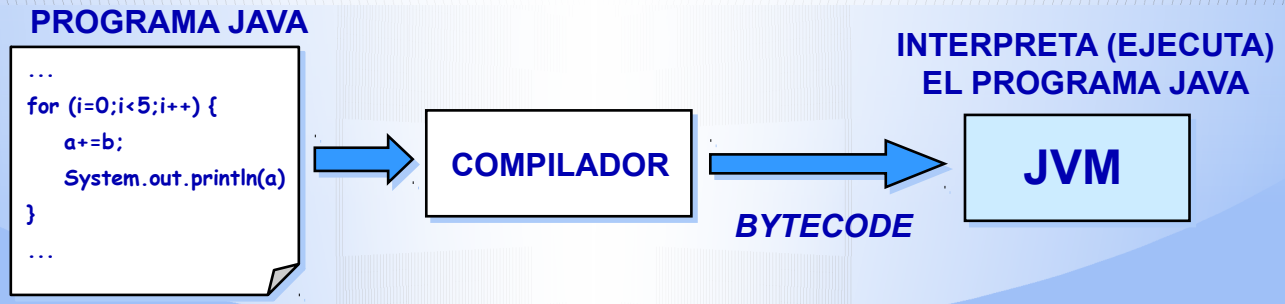
*Índice

- El lenguaje Java
- Clases y objetos
- Herencia
- Interfaces
- Excepciones

*El lenguaje Java

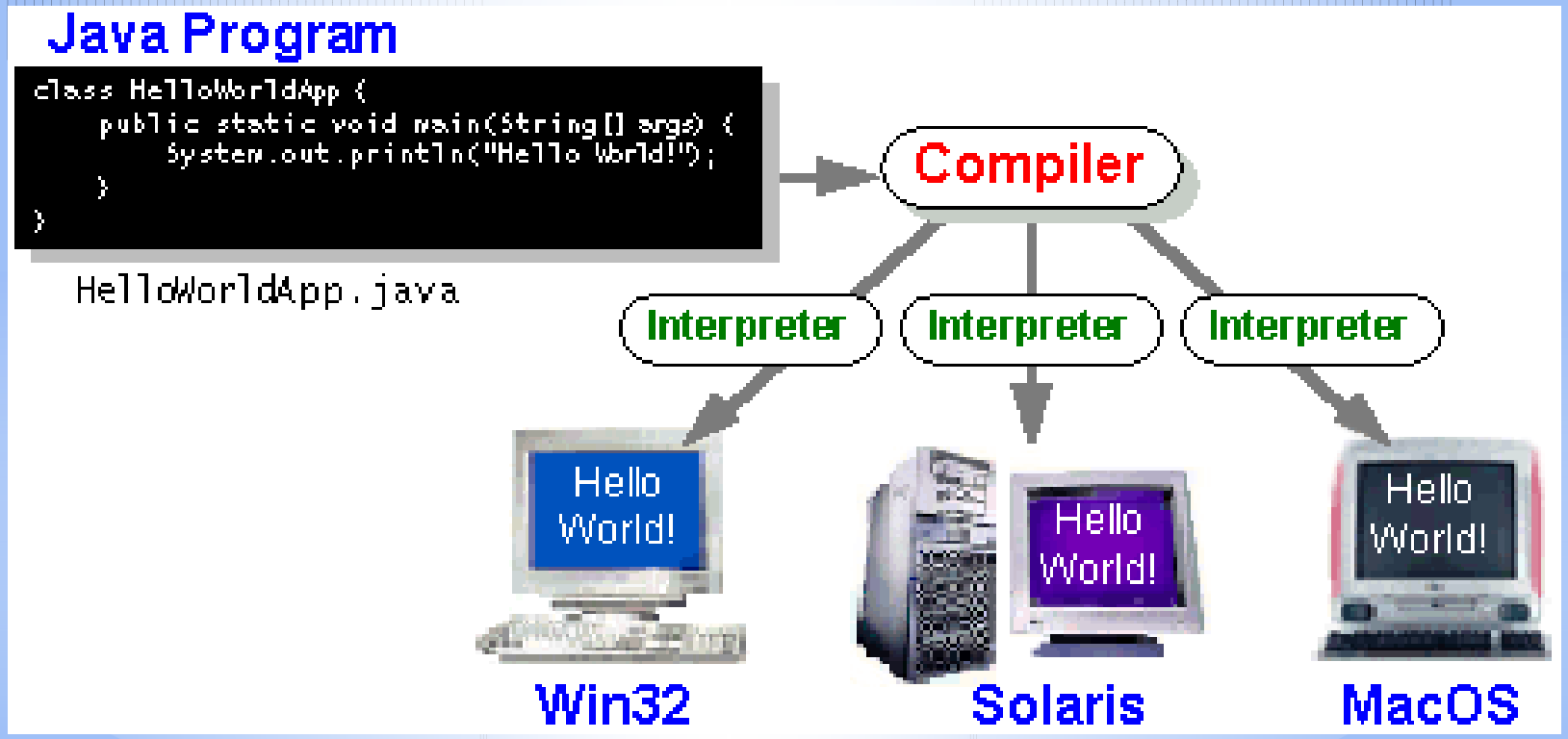
Características

- Lenguaje de programación de propósito general
- Orientado a objetos
- Lenguaje multiplataforma
- **Compilador** que traduce el código a un leguaje (***bytecode***)
- Se necesita un **intérprete Java** para ejecutar el *bytecode* en una máquina real



*El lenguaje Java

Características



*El lenguaje Java

Metodología de programación

- Analizar las especificaciones de problemas
  - Tener claro qué datos entran y qué datos salen
- Diseñar la solución
  - Organizar el problema en subproblemas
  - Identificar si se puede reutilizar código
- Implementar la solución
- Verificar la solución
- Mantener y actualizar el programa

*El lenguaje Java

Comentarios

- Es importante escribir un buen código
  - Fácil de comprender y modificar (por uno y por otros)
- Los **comentarios** son útiles para explicar qué hace el código

```
/* La clase HolaMundo imprime hola mundo en pantalla */
public class HolaMundo {
        public static void main(String[] args) {
                // Escribe ¡ Hola, Mundo !
                System.out.println("¡ Hola, Mundo !");
        }
}
```

*El lenguaje Java

Tipos y variables

- Los **tipos** de datos indican cómo se representa la información
  - Enteros (**byte**, **short**, **int**, **long**) en decimal, hexadecimal y octal
  - Reales (**float**, **double**)
  - **boolean**
  - **char** (entre comillas simples)

CUIDADO CON LAS MAYÚSCULAS Y MINÚSCULAS

- Se puede realizar conversión de tipos

- Tipos de datos complejos ⇒ **CLASES**
  - **String**

*El lenguaje Java

- Las **variables** son nombres (identificadores) que representan un dato (valor de cierto tipo)
- La declaración de una variable tiene el formato:

[modificador] tipo nombre_de_la_variable

- El contenido de una variable puede variar dentro del programa. Si no queremos que varíe se usa la palabra **final**
- Ejemplo:

```
int a;
char b='A';
double suma,resultado,c;
final double PI=3.1416;
```

Hay que darle valor

*El lenguaje Java

Vectores y matrices

- En Java se pueden declarar vectores y matrices de cualquier tipo

```
int v[];    // declaración de un vector de enteros
char b[][]; // declaración de una matriz de caracteres
v=new int[10]; // creación del vector de enteros anterior (tamaño 10)
b=new char[3][2]; // creación de la matriz de caracteres anterior (tamaño 3x2)
```

- Y su longitud:

```
v.length   // corresponde con el valor 10
b[0].length // corresponde con el valor 2
```

- Pueden crearse vectores y matrices de objetos

*El lenguaje Java

Operadores

Operador	Descripción
==	Igual que
!=	Distinto que
<=	Menor o igual que
<	Menor que
>=	Mayor o igual que
>	Mayor que

RELACIONALES

Operador	Descripción
&&	Y lógico
||	O lógico
!	No lógico

LÓGICOS

Operador	Descripción
+	Suma
-	Resta
*	Multiplicación
/	División
%	Resto de división entera
++	Incremento (i++ o ++i)
- -	Decremento (i- - o - -i)

ARITMÉTICOS

*El lenguaje Java

Lectura por teclado

- En Java se pueden leer desde teclado así:

```
/* Leemos datos desde teclado*/
import java.io.*;
public class HolaMundo{
	public static void main(String[] args) throw IOException {
		InputStreamReader isr = new InputStreamReader(System.in);
		BufferedReader br = new BufferedReader (isr);
		String frase = br.readLine();
		System.out.println("La frase leída es: " + frase);

	}
}
```

*El lenguaje Java

- En Java se pueden hacer conversiones de tipos de variables
  - String a Double

```
double numero = Double.parseDouble(cadena);
```

  - String a Float

```
float numero = Float.parseFloat(cadena);
```

  - String a Boolean

```
boolean booleano = Boolean.parseBoolean(cadena);
```

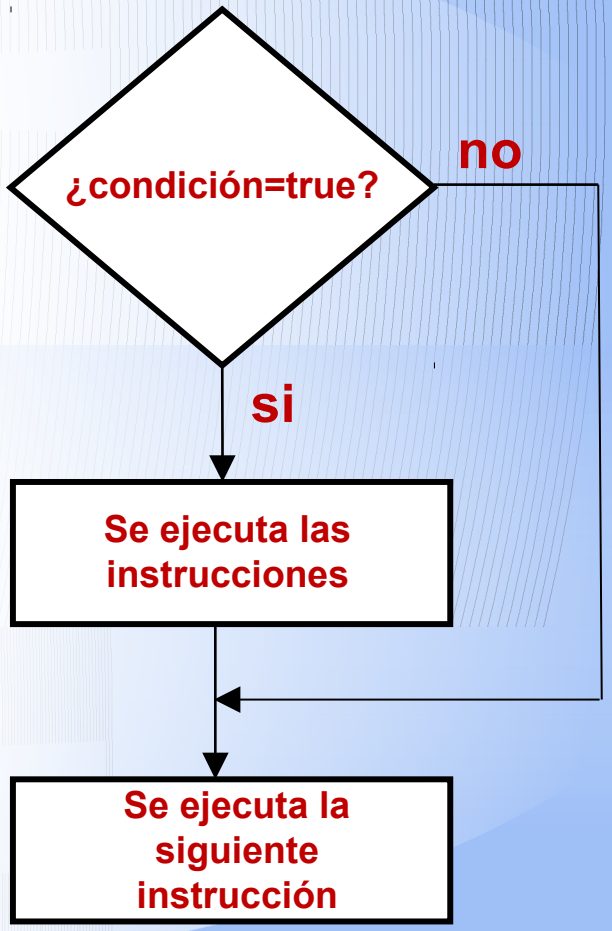
  - String a Entero

```
int numero = Integer.parseInt(cadena);
```

El lenguaje Java

Estructuras de control: condicional simple

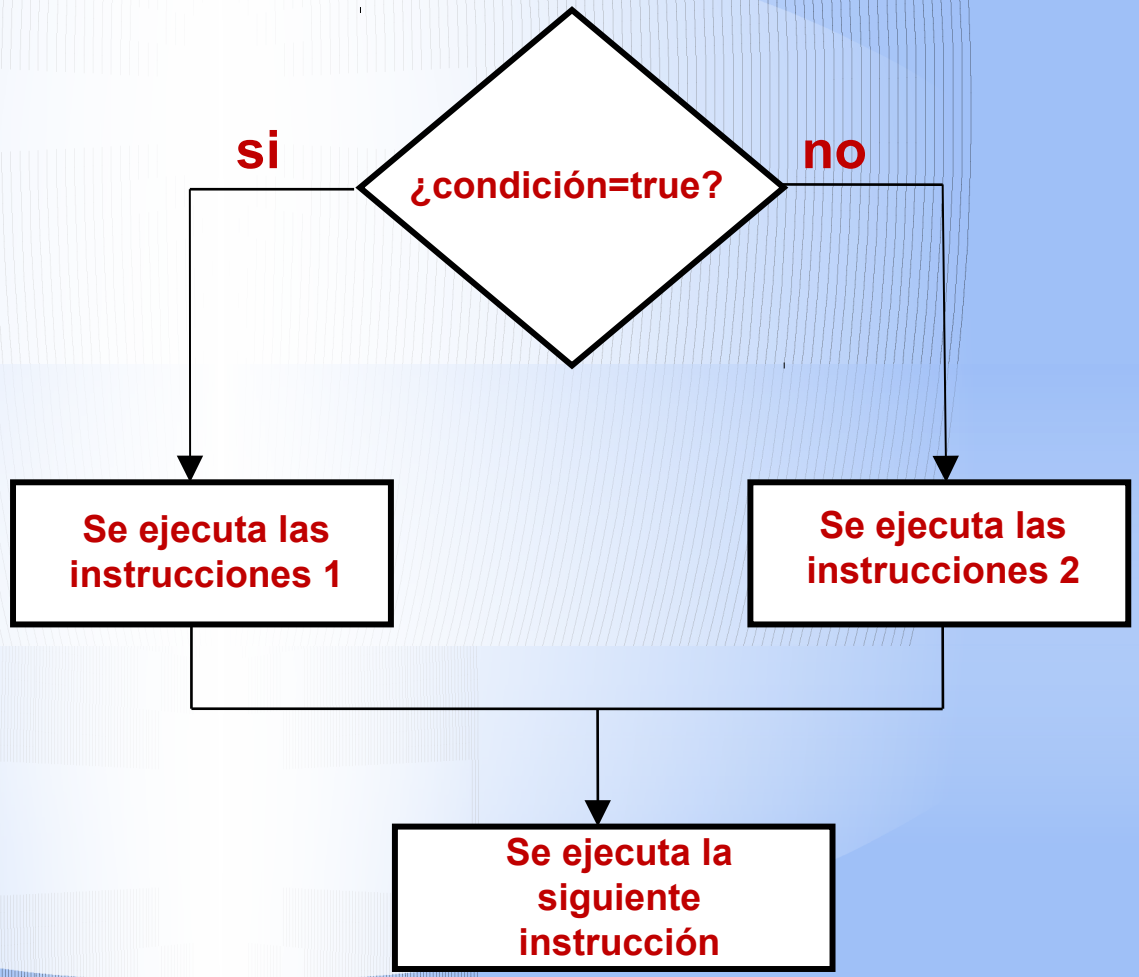
```
if (condición) {
      instrucción;
      …
}
siguiente instrucción;
```



*El lenguaje Java

Estructuras de control: condicional doble

```
if (condición) {
      instrucción 1;
      …
} else {
      instrucción 2;
      …
}
siguiente instrucción;
```

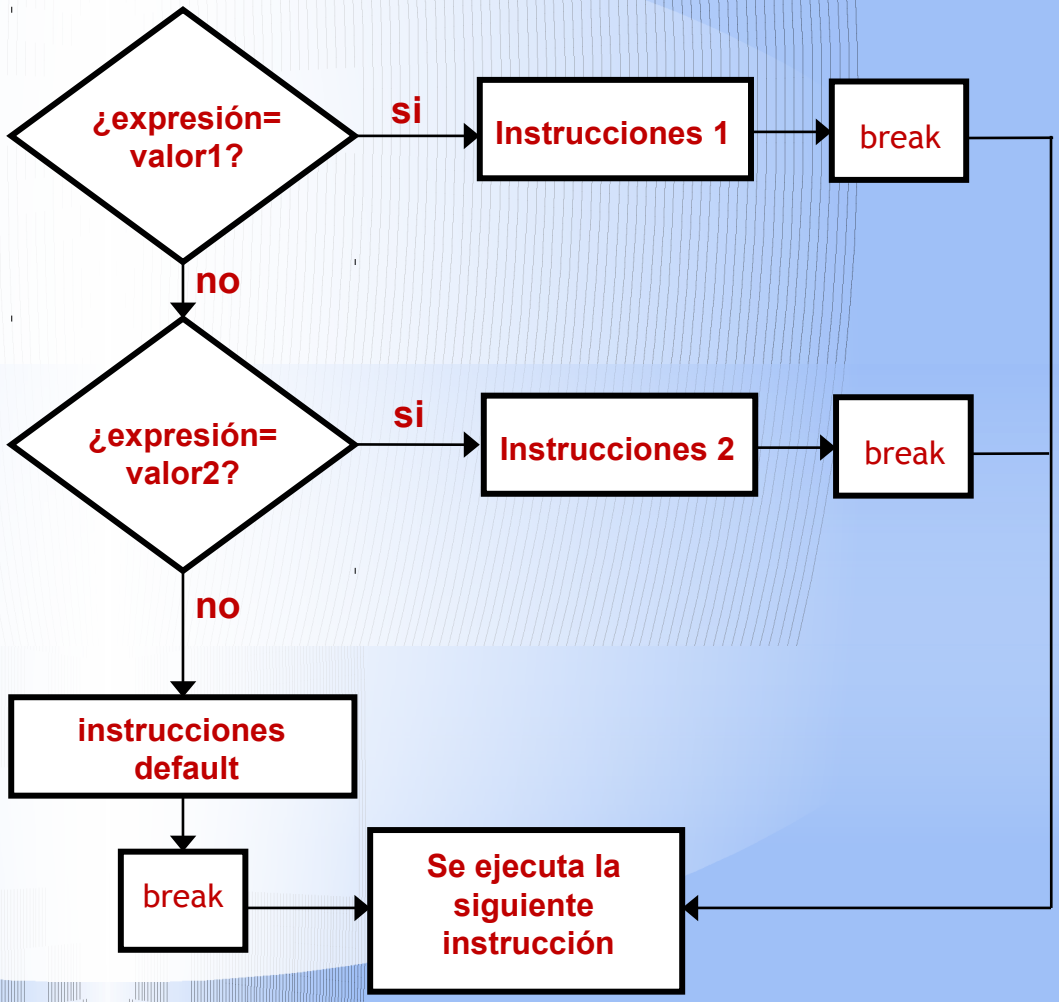


*El lenguaje Java

Estructuras de control: condicional múltiple

LA EXPRESIÓN SÓLO CHAR, INT, SHORT o BYTE

```
switch (expresión) {
    case valor1:
        // instrucciones 1;
        break;
    case valor2:
        // instrucciones 2;
        break;
    ...
    default:
        // instrucciones default
        break;
}
// siguiente instrucción
```

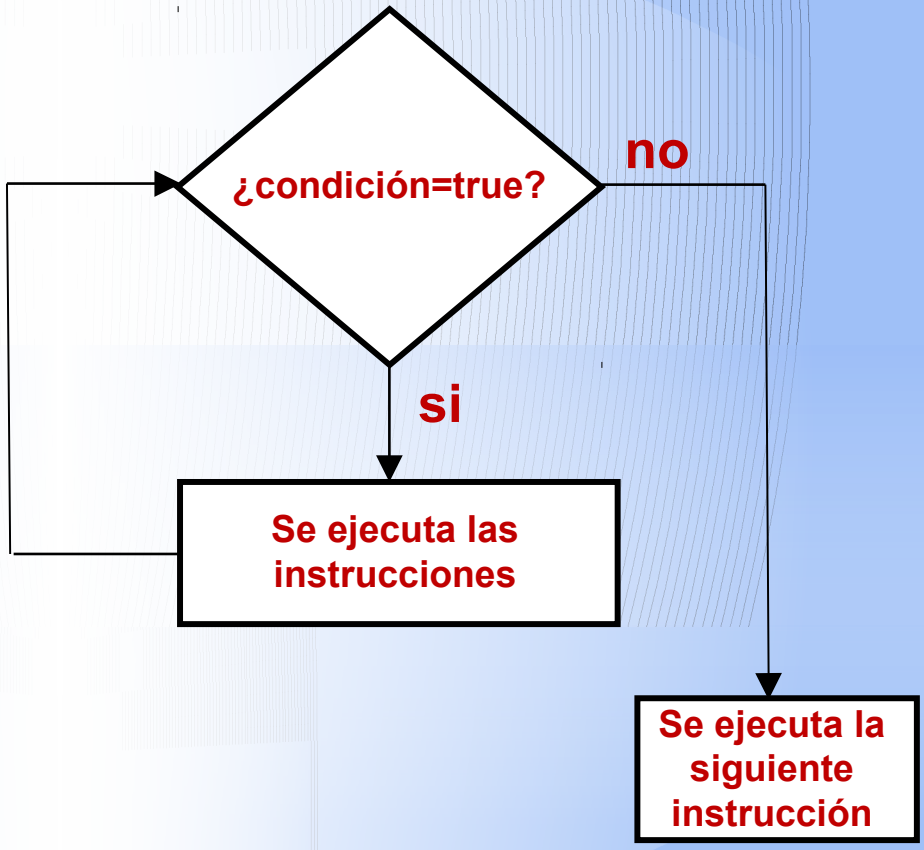


CUIDADO CON EL *break*

El lenguaje Java

Estructuras de control: bucle *while*

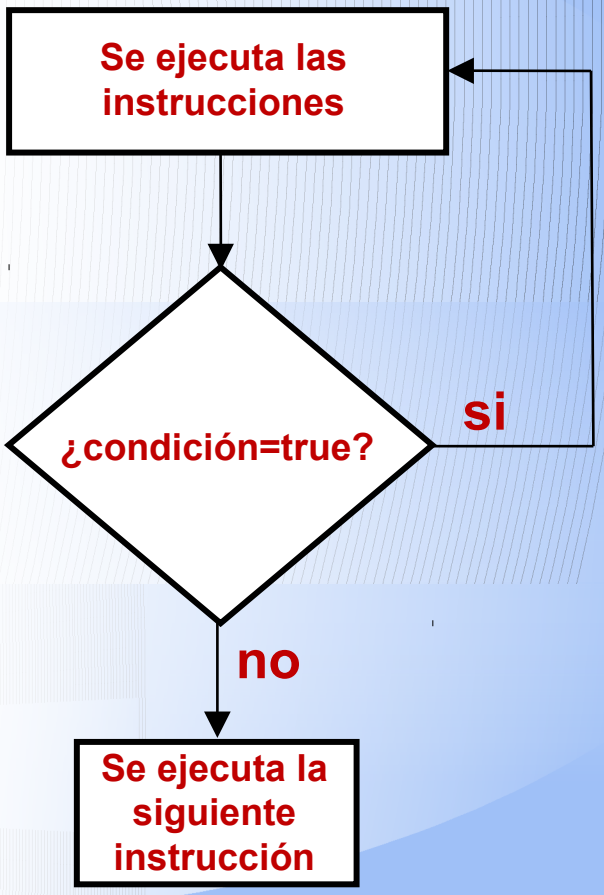
```
while (condición) {
      instrucción;
      …
}
siguiente instrucción;
```



*El lenguaje Java

Estructuras de control: bucle *do-while*

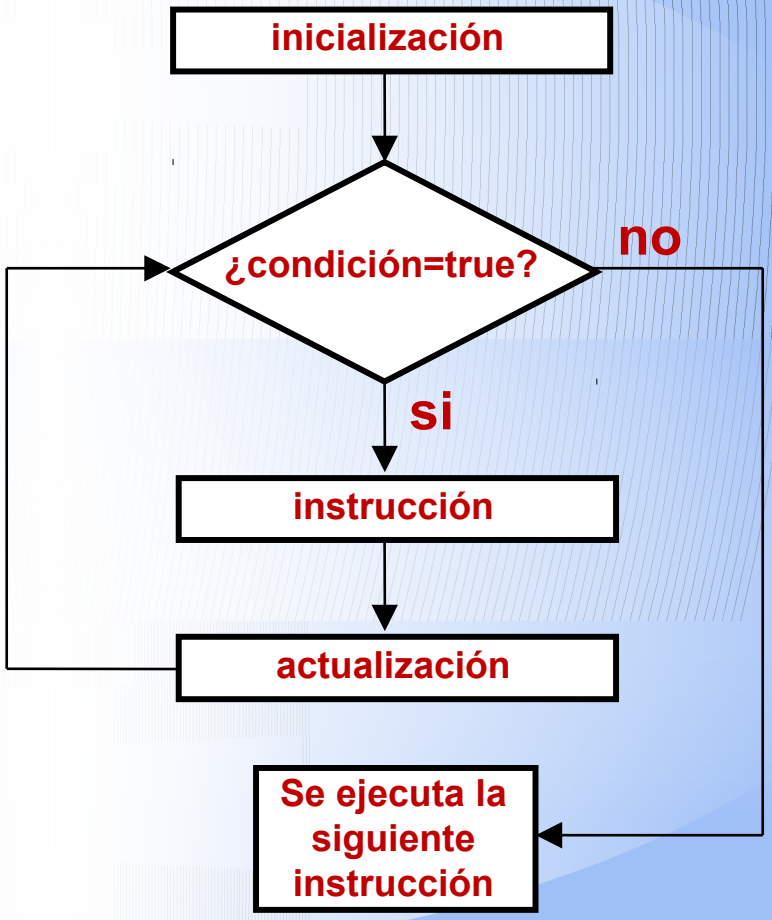
```
do {
      instrucción;
      …
} while (condición);
siguiente instrucción;
```



*El lenguaje Java

Estructuras de control: bucle *for*

```
for (inicialización;
      condición;
      actualización) {
     instrucción;
     …
}
siguiente instrucción;
```



*El lenguaje Java

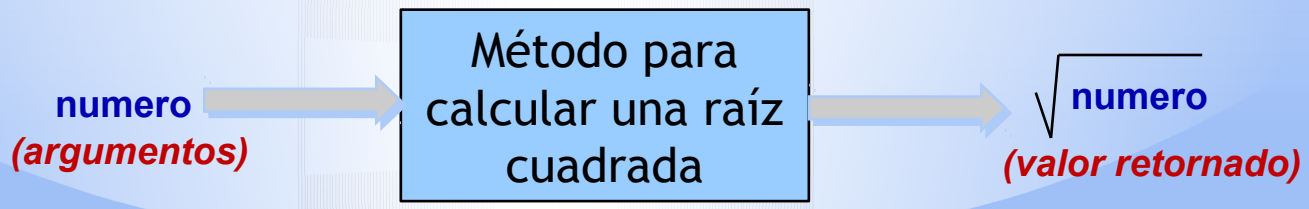
Métodos

- ¿Qué es un ***método***?
- ¿Cómo se declara?
  - La instrucción ***return***
- ¿Cómo se invoca?
- ¿Qué es el método ***main***?

*El lenguaje Java

¿Qué es un método?

- Es una forma de organizar las instrucciones dentro de una unidad con un nombre dado
  - Es lo que conocemos como procedimiento o función
- Se caracteriza por:
  - Permite la reutilización de código
  - El programa es más fácil de comprender y depurar
  - Puede aceptar distintas entradas (***argumentos***)
  - Permite la salida de un valor



*El lenguaje Java

¿Cómo se declara?

- Un método tiene cuatro partes:
  - El **tipo** retornado (primitivos, no primitivos, ***void***)
  - El **nombre** (identificador)
  - Los **argumentos**
  - El **cuerpo**

```
double sqrt(double numero) {
    } /* operaciones que realizan el cálculo
    de la raíz cuadrada de un número */
```

- El tipo, el nombre y los argumentos son la *firma* del método
- Métodos con el mismo nombre deben tener distinta firma (*sobrecarga*)

*El lenguaje Java

La instrucción *return*

- La instrucción ***return*** se usa para devolver el resultado de la ejecución del método
- El formato es el siguiente:

return valor o expresión;

- El tipo del valor o expresión retornada debe ser el mismo que el tipo del método

```
double sqrt(double numero) {
        double resultado;
        /* operaciones que realizan el cálculo
        de la raíz cuadrada de un número */
        return resultado;
}
```

*El lenguaje Java

¿Cómo se invoca a un método?

- Para llamar a un método se indica su nombre y sus argumentos, si los tiene, entre paréntesis y separados por comas

```
sqrt(numero);
```

- Si el método no tiene argumentos, se llama igual pero entre paréntesis no hay nada

```
limpiarPantalla();
```

- Si el método retorna un valor, ese valor debería guardarse

```
double resultado=sqrt(numero);
```

- La forma de llamar a un método también depende de dónde se llame a éste

*El lenguaje Java

El método *main*

- Es el método usado por Java para empezar la ejecución de un programa
- Tiene un firma fija (no puede cambiar)

```
public static void main(String[ ] args)
```

- El contenido del cuerpo del *main* dependerá de la implementación del programador
- Se puede introducir datos al programa desde la línea de órdenes a través del argumento *args*

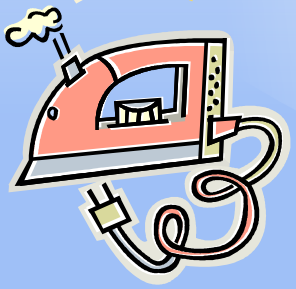
*Índice

- El lenguaje Java
- Clases y objetos
- Herencia
- Interfaces
- Excepciones

*Clases y objetos

¿Qué es un objeto?

- Los objetos son representaciones de cosas (del mundo real o imaginario) y pertenecen a una clase determinada
- Los objetos tienen características o atributos como: el color, la altura, la temperatura, la velocidad, entre otros
- Los objetos tiene un comportamiento o métodos



*Clases y objetos

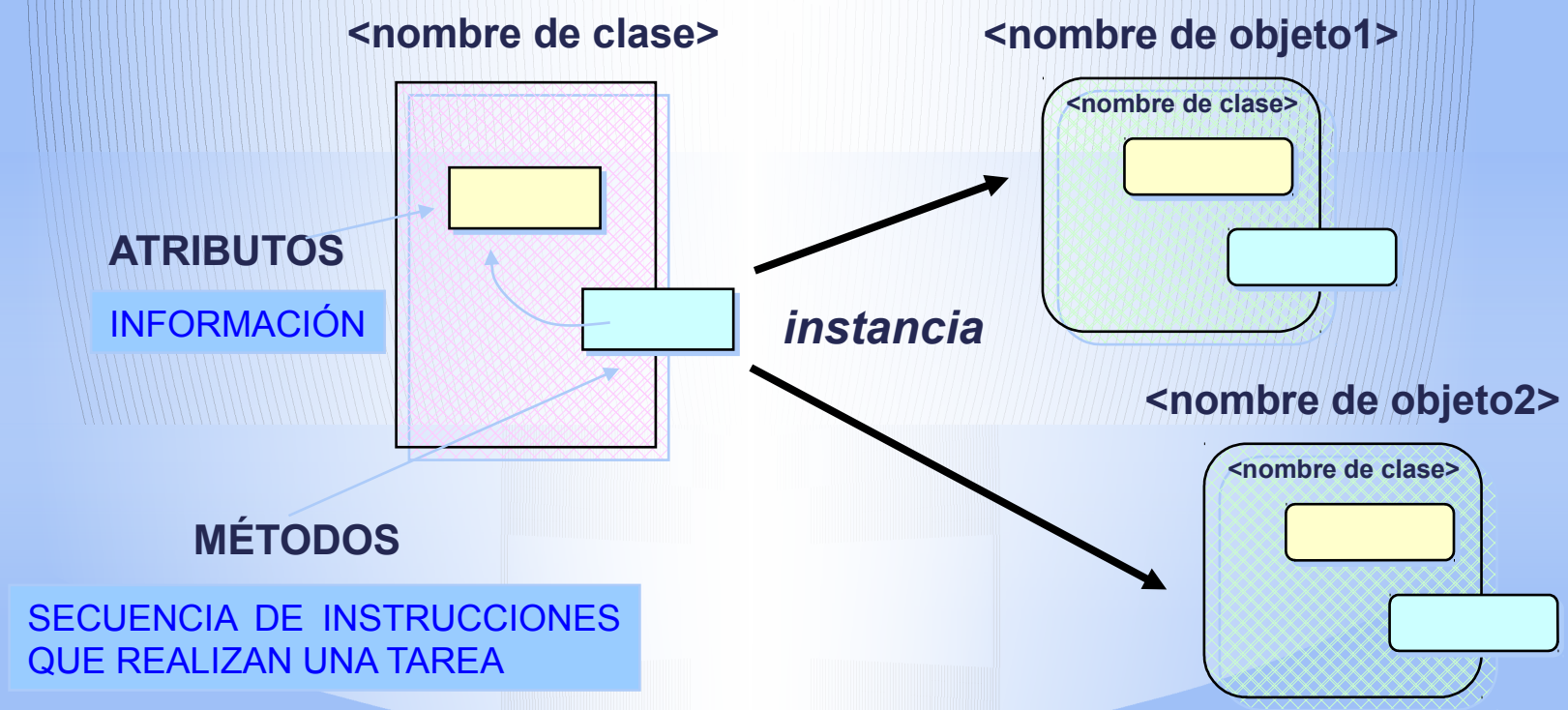
¿Qué es un objeto?

- Los objetos tienen miembros públicos (**public**) y miembros privados (**private**)
- En la programación orientada a objetos, la mayoría de los miembros son privados, existiendo algún miembro público para que se puede interactuar con ellos desde el exterior (ocultación de la información)
- Normalmente, los miembros privados de un objeto son los atributos (variables) y los miembros públicos son los métodos que acceden a ellos

*Clases y objetos

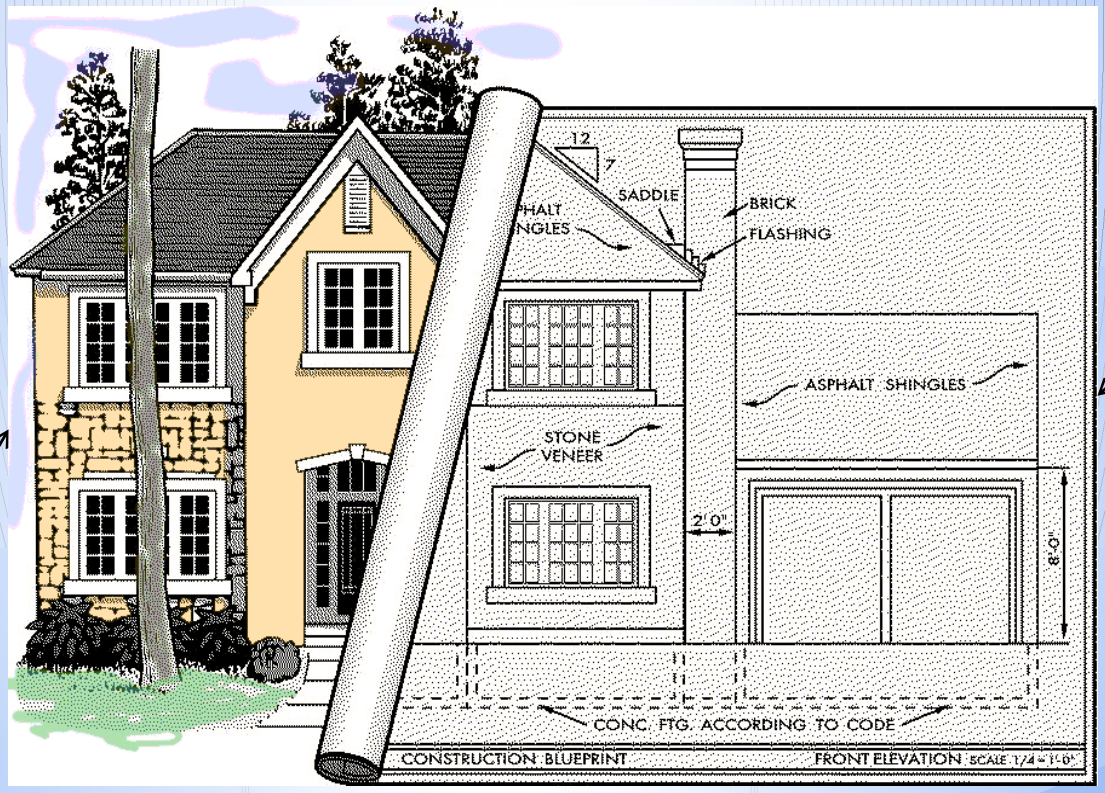
¿Qué es una clase?

- Las clases son tipos de datos, es decir, plantillas para crear objetos (indicando sus atributos y métodos)



*Clases y objetos

¿Qué es una clase?



clase

objeto

*Clases y objetos

Clase *vs* objeto

CLASE

- Son tipos abstractos de datos
- Definen el comportamiento y los atributos de un grupo de objetos, si éstos fueran creados a partir de la clase

Clase Transporte

Métodos: avanzar, frenar, acelerar, …
Atributos: color, velocidad, longitud, …

OBJETO

- Es un ejemplo o instancia de una clase (existe en la memoria del ordenador)

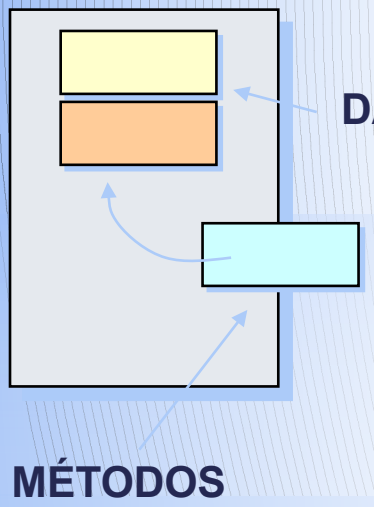
Objeto guagua
pertenece a la clase Transporte

Objeto guagua

Clase Transporte
Métodos: avanzar, frenar, acelerar, …
Atributos: color="Amarillo"
velocidad=150 Km/h …

*Clases y objetos

Declaración de clases



public
final
abstract

```
[modificadores] class <nombre_clase>
[extends <nombre_superclase>]
[implements <nombre_inferface1>,<nombre_interface2>,...] {
    // variables (de instancia)
    // variables (estáticas)
    // métodos
}
```

*Clases y objetos

Declaración de clases

```
import <nombre de paquetes o clases>;

class <nombre de clase> {
        …
        …
}
…
public class <nombre de clase> {
        …
        public static void main(String[] arg) {
                …
        }
}
```

CLASES PREDEFINIDAS

<nombre de paquete>.<nombre de clase>

CLASES DEFINIDAS POR EL PROGRAMADOR

CLASE PRINCIPAL

- En un archivo sólo puede haber una clase pública
- El nombre del archivo coincide con el de la clase pública
- La clase raíz en Java es la clase **Object** que está en **java.lang**

*Clases y objetos

Clases de variables

LOCALES	DECLARADA DENTRO DE UN MÉTODO Y SÓLO SE CONOCE EN EL MÉTODO
DE INSTANCIA	DECLARADA DENTRO DE UNA CLASE (FUERA DE SUS MÉTODOS). **PERTENECE A LOS OBJETOS**
DE CLASE O ESTÁTICAS	DECLARADA DENTRO DE UNA CLASE (FUERA DE SUS MÉTODOS). **PERTENECE A LA CLASE**

VARIABLES MIEMBROS

static { ***public*** / ***private*** / ***protected*** / **sin modificador** }

[modificador] tipo nombre_variable;

primitivos / ***no primitivos***

*Clases y objetos

Clases de variables

```
public class MiClase {
    private float a,b;
    public boolean var;
    private static int suma;
    public static double
    total;
    protected char opcion;
    protected static String
    dato;

    int paso;

    ...
```

Variables de instancia:

a, b, var, opcion, paso

Variables estáticas:

suma, total, dato