Universidad
Carlos III de Madrid

Tema 1

Introducción a la Programación

Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- Breve introducción histórica a la programación
- Compilación vs. interpretación de programas
- Paradigmas de programación
- Programación orientada a objetos: Java
- Resumen y Referencias

¿Qué es programar?

- Según la RAE:
  - **5.** tr. *Inform.* Elaborar programas para la resolución de problemas mediante ordenadores

- Una definición informal pero más explicativa:
  - Proporcionar a un ordenador un conjunto de datos y unas instrucciones sobre lo que se debe hacer con esos datos con el objetivo de resolver algún problema

¿Qué es programar?

Objetivo: Resolver un Problema

Cómo resolverlo: Utilizando un **algoritmo** y los **datos** del problema.

- Un **algoritmo** es:
  - Un conjunto de instrucciones que en una determinada secuencia permite la resolución de un problema paso a paso.
  - Una lista bien definida, ordenada y finita de operaciones que permite hallar la solución a un problema.
- Los **datos** son los propios del problema y serán manejados por el algoritmo

Ejemplo de algoritmo

- Algoritmo para cambiar la rueda de un coche

Datos: rueda pinchada y ubicación del gato, de la rueda de repuesto y de la llave inglesa

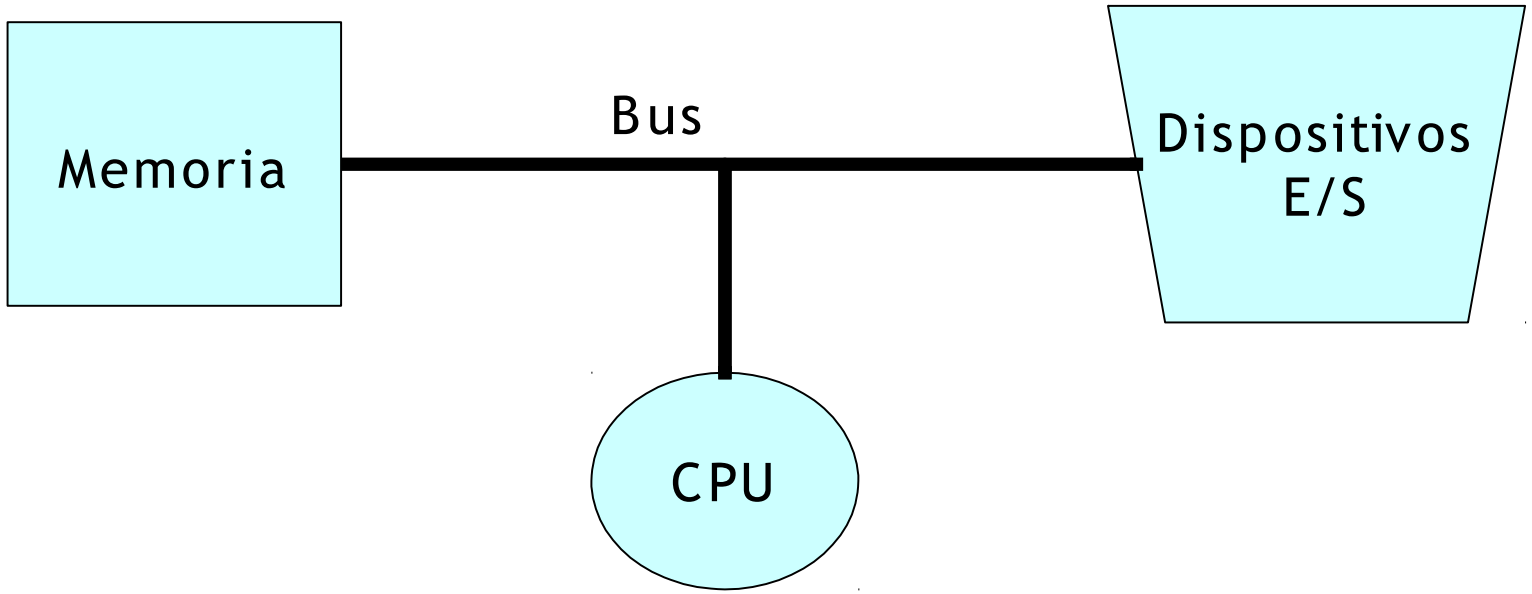
- PASO 1. Aflojar los tornillos de la rueda pinchada con la llave inglesa
- PASO 2. Colocar el gato mecánico en su sitio
- PASO 3. Levantar el gato hasta que la rueda pinchada pueda girar libremente
- PASO 4. Quitar los tornillos
- PASO 5. Quitar la rueda pinchada
- PASO 6. Poner rueda de repuesto
- PASO 7. Poner los tornillos y apretarlos ligeramente
- PASO 6. Bajar el gato hasta que se pueda liberar
- PASO 7. Sacar el gato de su sitio
- PASO 8. Apretar los tornillos con la llave inglesa

Agenda

- ¿Qué es programar?
- **Arquitectura básica de un ordenador**
- Breve introducción histórica a la programación
- Compilación vs. interpretación de programas
- Paradigmas de programación
- Programación orientada a objetos: Java
- Resumen y Referencias

Arquitectura básica de un ordenador

- Hardware vs. Software
- El 99% de los ordenadores actuales tiene la siguiente arquitectura[1]:



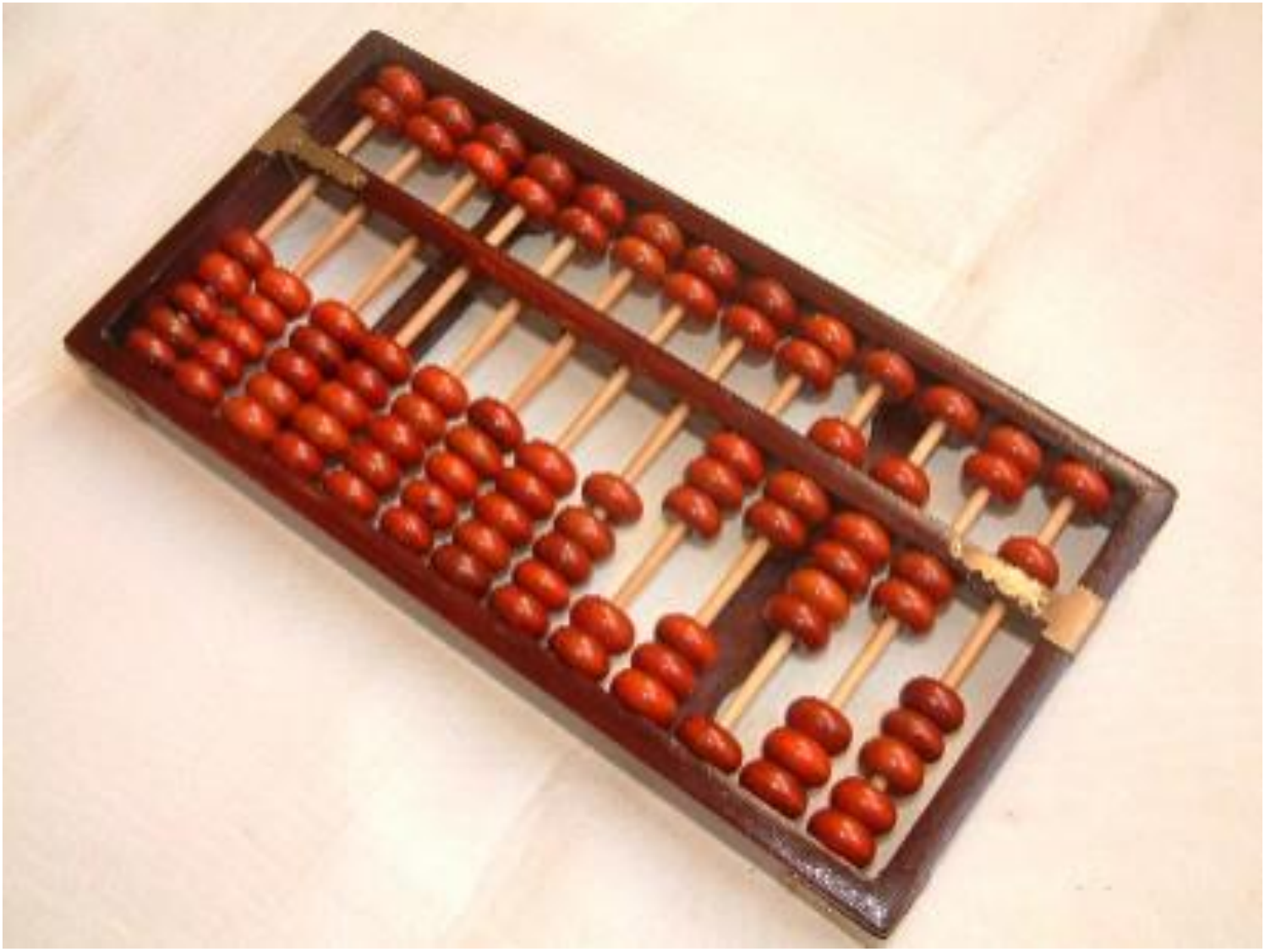
- Datos e instrucciones comparten memoria

[1]Denominada arquitectura de Von Neumann, aunque fue propuesta inicialmente por Eckert y Mauchly

Características de la arquitectura

- Componentes:
  - Unidad central de proceso (CPU)
    - Se encarga fundamentalmente de ejecutar las instrucciones y coordinar el resto de elementos
  - Memoria
    - Almacena los datos, las instrucciones y los resultados
    - Clasificación: principal/secundaria, permanente/volátil, acceso directo/secuencial
  - Dispositivos de entrada/salida
    - Para proporcionar los datos e instrucciones y recibir los resultados
  - Bus de datos
    - Para compartir la información entre los componentes anteriores

Ejemplo de ordenador real



Tipos de software

- Básicamente se puede dividir en 2 tipos:
  - Software de **Sistema** (Sistema operativo)
    - Proporciona control sobre el hardware y sirve de base a las aplicaciones
  - Software de **Aplicaciones**
    - Programas con finalidades específicas, resuelven un problema o familia de problemas determinados
      - Ofimática (procesadores de texto, hojas de cálculo…)
      - Contabilidad
      - Control
      - Juegos
      - …

Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- **Breve introducción histórica a la programación**
- Compilación vs. interpretación de programas
- Paradigmas de programación
- Programación orientada a objetos: Java
- Resumen y Referencias

Lenguaje de programación

- Programa = datos + instrucciones
- Para comunicarle al ordenador el programa se usa un lenguaje de programación
- Los ordenadores no entienden lenguaje natural
  - **¿Cómo decimos al ordenador lo que tiene que hacer?**: Escribiendo un programa en un lenguaje de programación determinado, para implementar ese algoritmo
- Lenguajes específicos vs. generales

Tipos de lenguajes de programación

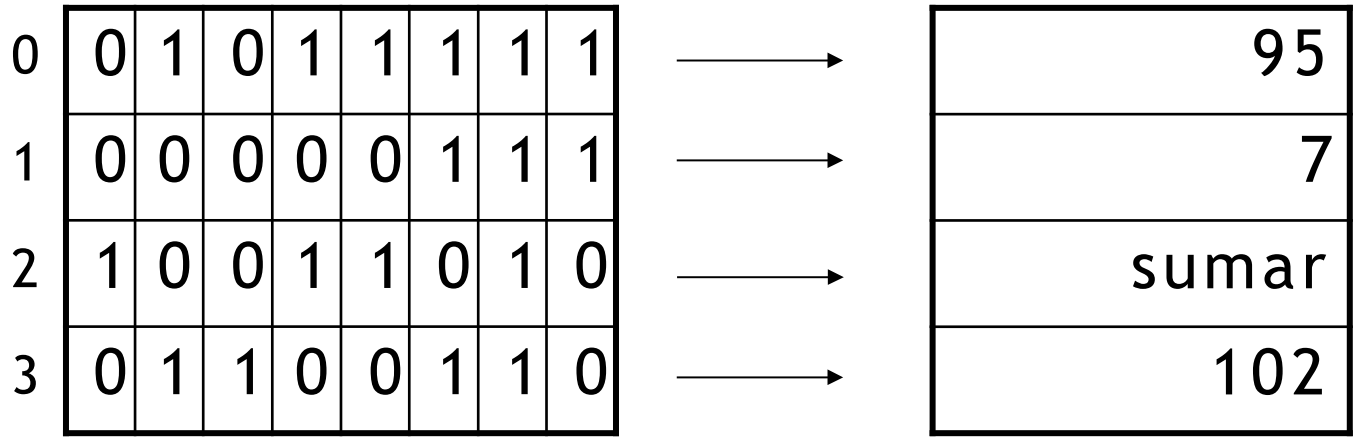
- Lenguaje **Binario o Código Máquina**
  - 0 y 1
- Lenguajes de **Bajo Nivel**
  - Instrucciones básicas (mover datos, sumar…)
- Lenguajes de **Alto Nivel**
  - Más cercanos al lenguaje natural
  - … aunque tampoco demasiado

Código Máquina

- Lenguaje Binario o Código Máquina: único lenguaje que entiende el ordenador
  - Datos e instrucciones se codifican mediante conjuntos de 0 y 1
  - El más rápido: hablamos al ordenador en su propio lenguaje
  - Muy propenso a errores, muy complicado

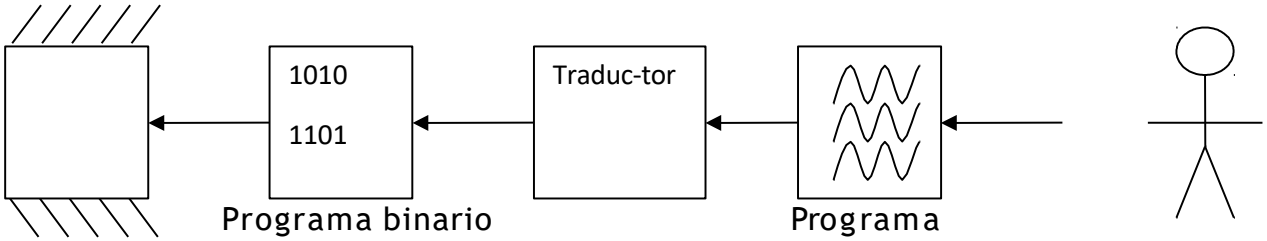
Representación de datos e instrucciones

- La memoria está compuesta por bits que sólo pueden valer 1 o 0
  - Información codificada en binario
- Los bits se agrupan en bytes (8 bits)
- Cada celda de memoria contiene entre 1 y 8 bytes y almacena un dato, un resultado o una instrucción



Lenguajes de bajo nivel

- Se usa un traductor para convertir desde un lenguaje textual a código máquina
  - El traductor es un programa que le dice al ordenador cómo realizar la traducción



  - Nace el lenguaje ensamblador: cambiamos 0 y 1 por texto, pero sigue siendo muy poco intuitivo
    - Depende totalmente del tipo de procesador

Lenguajes de bajo nivel

- Ejemplo de código en ensamblador

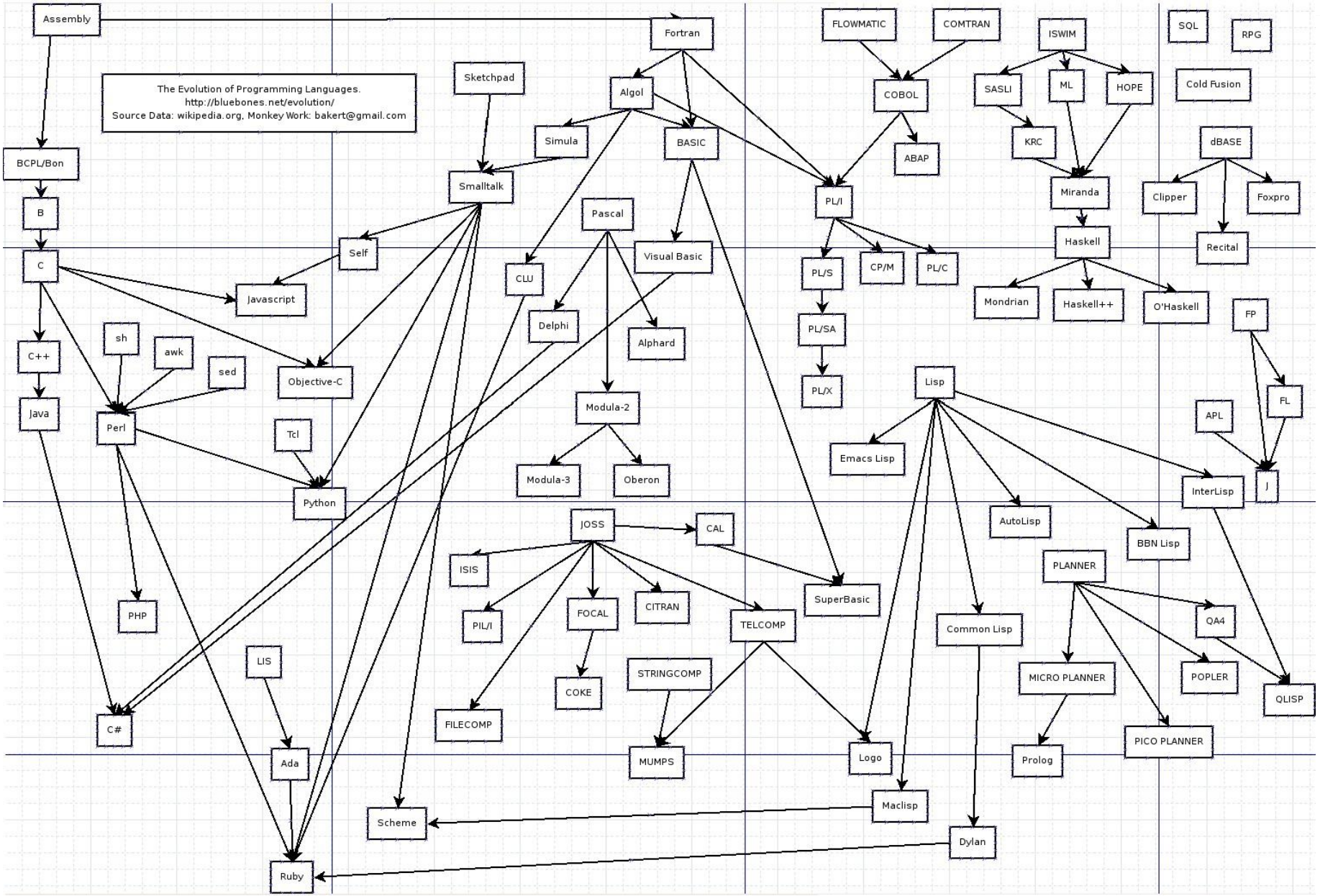
```
.model small
.stack
.data
Cadena1 DB 'Hola Mundo.$'
.code

programa:
   mov ax, @data
   mov ds, ax
   mov dx, offset Cadena1
   mov ah, 9
   int 21h
   mov  ah,4ch
   int 21h
end programa
```

Lenguajes de alto nivel (I)

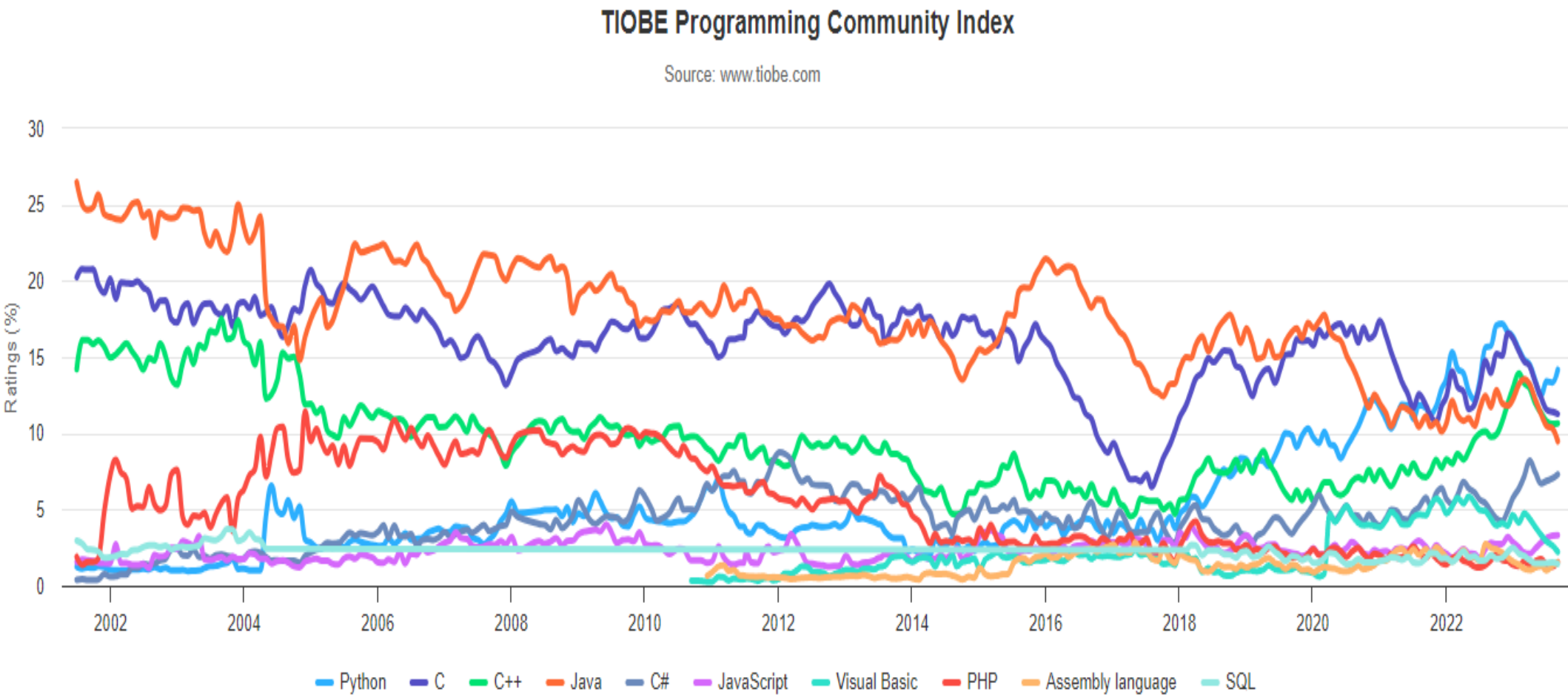
- Intentan acercar el lenguaje de programación al lenguaje humano
  - Luego el ordenador se encargará de traducir
  - Ideal: poder usar lenguaje natural
- Existen más de 300 (unos 2400 con dialectos)
- Los pioneros incluían conceptos como:
  - Variables: no es necesario gestionar los datos directamente en la memoria
  - Estructuras de datos complejas
  - Nuevas instrucciones, distintas de las que proporciona el ordenador
- Historia:

  http://manuelpereiragonzalez.blogspot.com/2009/09/historia-de-la-informatica-los.html

La evolución de forma gráfica



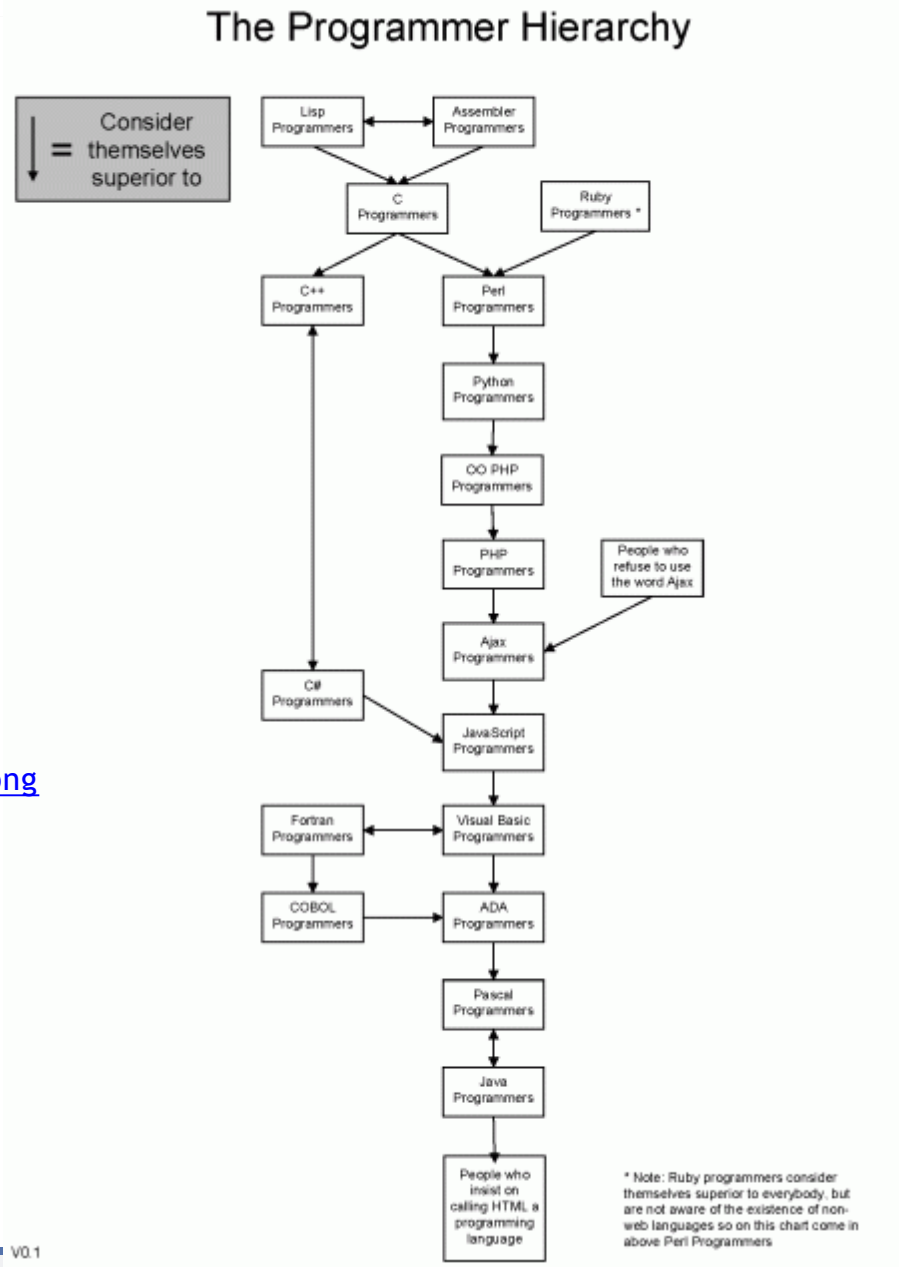
Uso de algunos lenguajes



Jerarquía de los programadores

Quién se considera superior a quién

http://www.hermann-uwe.de/files/images/programmer_hierarchy.png



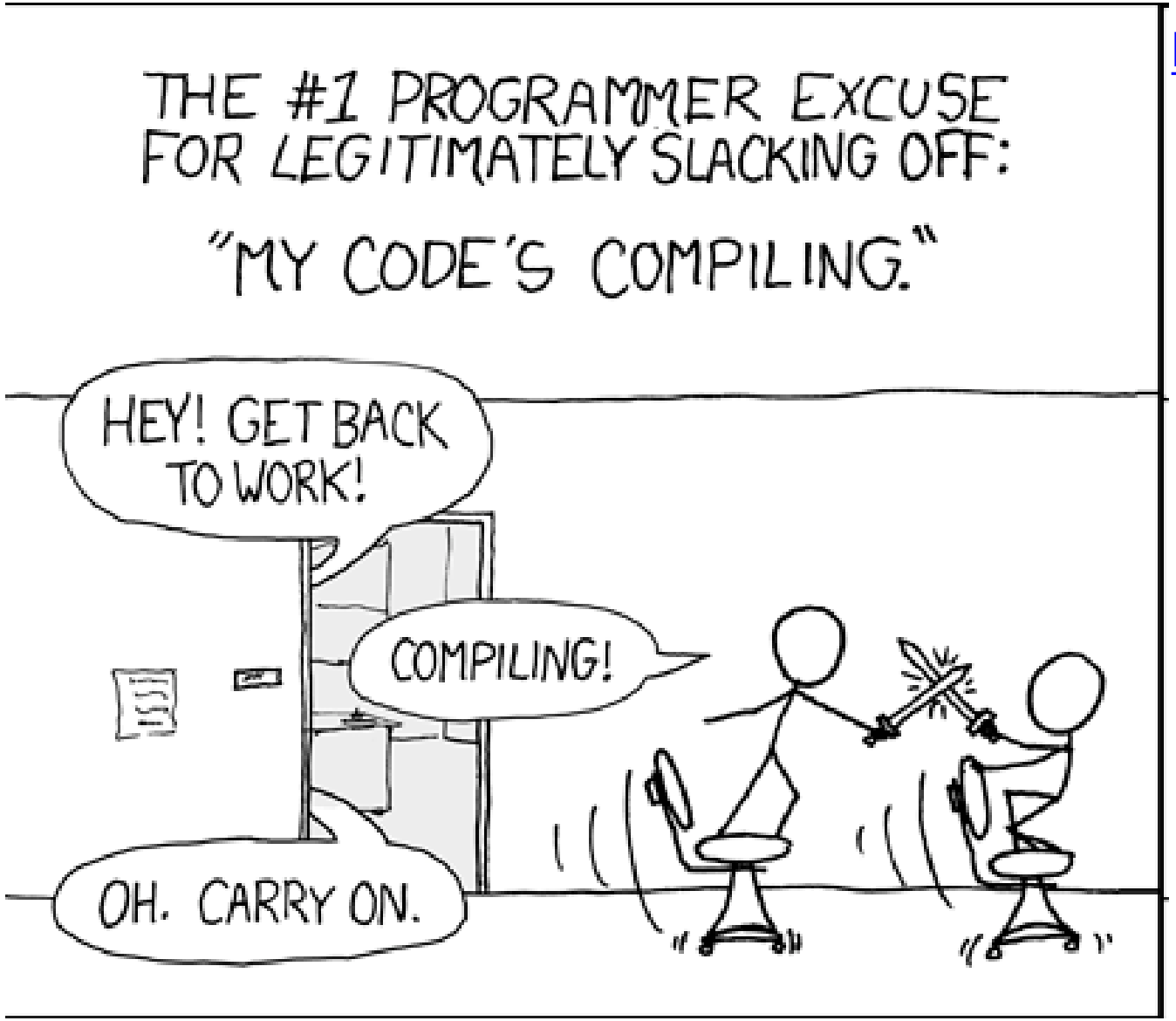
Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- Breve introducción histórica a la programación
- **Compilación vs. interpretación de programas**
- Paradigmas de programación
- Programación orientada a objetos: Java
- Resumen y Referencias

Compilación e interpretación

- La traducción desde un lenguaje de programación a binario se puede hacer de dos formas:
  - Todo a la vez: compilador
    - Ejecución más rápida
  - Instrucción a instrucción: intérprete
    - Ejecuta aunque haya errores en el código
    - Permite cambios “en caliente”

Compilación



http://xkcd.com/303/

Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- Breve introducción histórica a la programación
- Compilación vs. interpretación de programas
- **Paradigmas de programación**
- Programación orientada a objetos: Java
- Resumen y Referencias

Paradigmas de programación

- Tres formas principales de darle las instrucciones al ordenador:
  - Programación **imperativa** (Java, C, C++, Python)
    - Se describen los pasos necesarios para solucionar el problema
  - Programación **funcional** (Lisp, Haskell, Erlang,F#)
    - Las instrucciones se dan mediante funciones "matemáticas" que transforman los datos.
  - Programación **lógica** (Prolog)
    - Se describe el problema pero no se dan instrucciones: se resuelve mediante inferencia lógica.
- Ninguno es siempre mejor que otro
- Muchos lenguajes son mixtos
  (ver http://www.info.ucl.ac.be/~pvr/paradigmsDIAGRAMeng.pdf)

Paradigmas de programación

Java – *Factorial.java*

```
public class Factorial {
  public static double factorial(int n) {
    int f = 1;
    for (int i = 2; i <= n; i++) f *= i;
    return f;
  }

  public static void main(String[] args) {
    factorial(42);
  }
}
```

Haskell – *fac.hs*

```
fac 0 = 1 fac n = n * fac (n-1)
main = print (fac 42)
```

Prolog - *factorial.hs*

```
factorial(0,1).
factorial(N,F) :-
N > 0,
N1 is N-1,
factorial(N1,F1),
F is N * F1.
?- factorial(42,X).
```

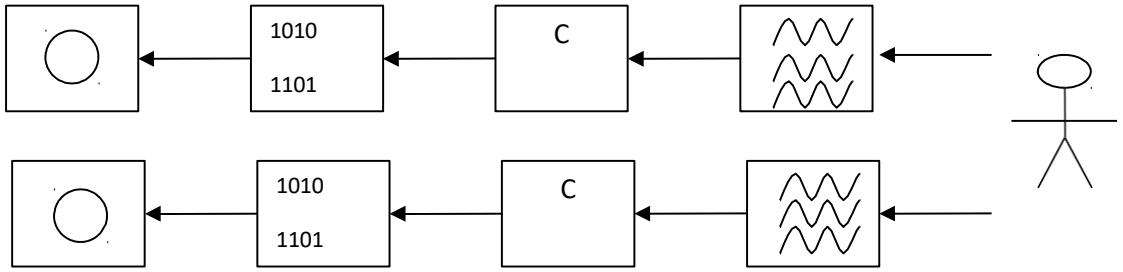
Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- Breve introducción histórica a la programación
- Compilación vs. interpretación de programas
- Paradigmas de programación
- **Programación orientada a objetos: Java**
- Resumen y Referencias

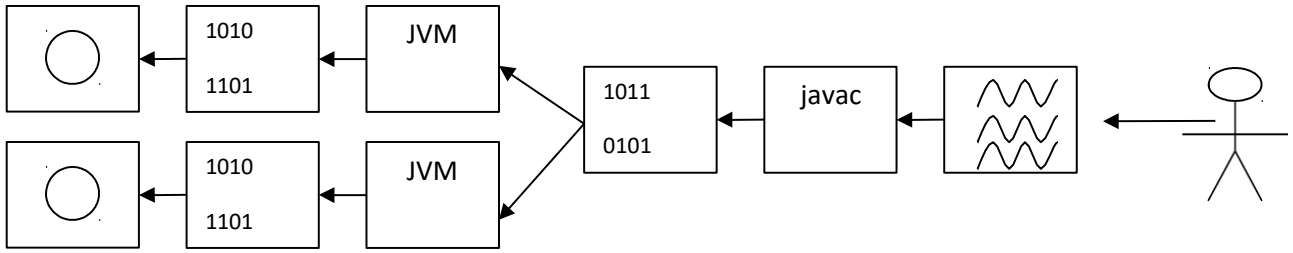
¿Qué es Java?

- Lenguaje de programación de alto nivel orientado a objetos
  - Es también una plataforma de desarrollo
- 1991: Sun Microsystems diseña un lenguaje para sistemas embebidos, (set-top-boxes), electrodomésticos
  - Lenguaje sencillo, pequeño, neutro
  - Necesidad de un nuevo lenguaje:
    - Orientado a objetos
    - Multiplataforma
  - Ninguna empresa muestra interés por el lenguaje
- Java: tipo de café

Historia de Java (I)

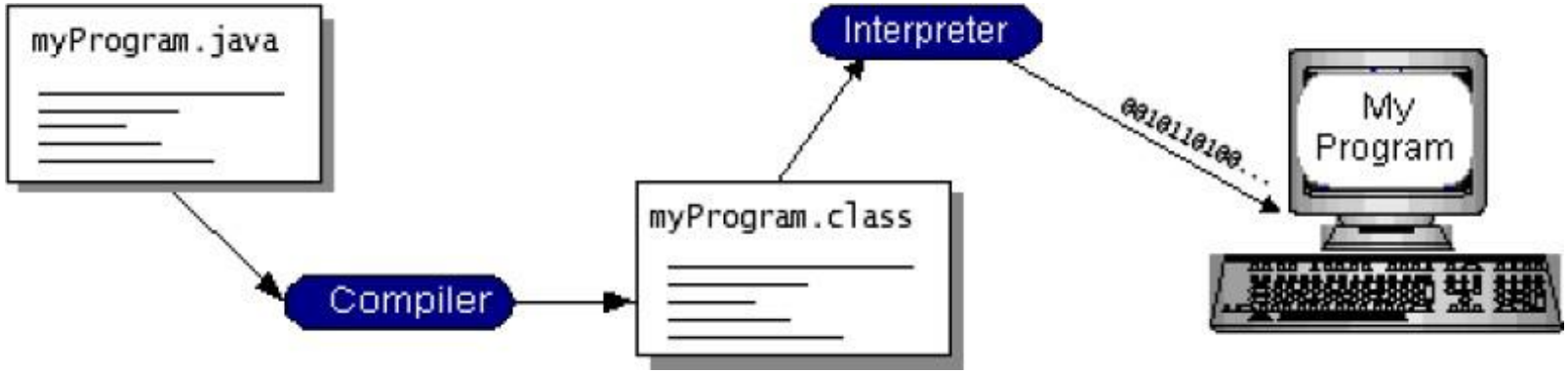


- Programas ligeramente distintos para distintas lavadoras
- Solución: lavadora virtual -> javalavadora
- Ganancia: un solo programa, aunque haya que hacer tres cosas (2 JVM y un compilador)



Compilado e interpretado

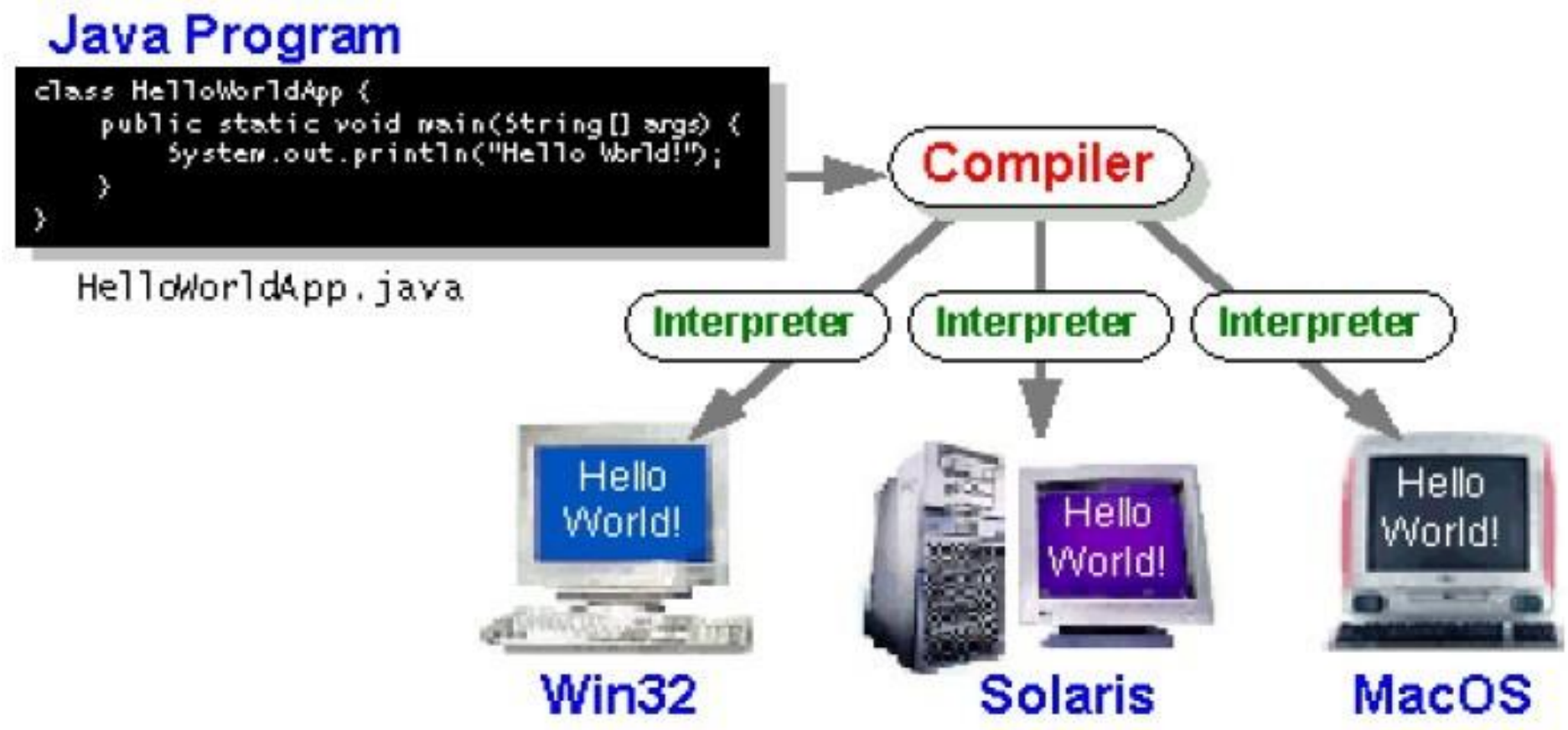
El lenguaje de programación Java es inusual por el hecho de que un programa a la vez se compila e interpreta



Con el compilador, un programa se traduce a un lenguaje intermedio llamado *Java bytecodes* —*estos códigos son independientes de la plataforma*— que será interpretado por el intérprete en la plataforma Java

Compilado e interpretado

"Escribir una vez, ejecutar en cualquier sitio". Un programa ***.java*** puede compilarse en cualquier plataforma que tenga un compilador Java. El fichero con los códigos de bytes ***.class*** puede entonces ejecutarse en cualquier implementación de la VM de Java.



Historia de Java (II)

- 1995: Java se presenta como lenguaje para Internet
- Netscape 2.0 introduce la primera JVM en un navegador web
- Filosofía Java: **"Write once, run everywhere"**
- 1997: Aparece Java 1.1. Muchas mejoras respecto a 1.0
- 1998: Java 1.2 (Java 2). Plataforma muy madura
- Apoyado por grandes empresas: IBM, Oracle, Inprise, Hewlett-Packard, Netscape, Sun
- 1999: Java Enterprise Edition. Revoluciona la programación en el lado servidor

Características Principales de Java

- Orientado a Objetos
- Totalmente Portable
- Lenguaje Interpretado (compilado a código intermedio, no a código máquina)
  - Java Virtual Machine (JVM)
  - ByteCode: Independiente de la máquina
- Gestión Automática de Memoria Dinámica
  - Recolector de basura (Garbage Collector)
- Sensible a Mayúsculas / Minúsculas
- Distribuido
- Robusto
- ¿Seguro?
- ¿Lento?

Versiones

- 1.0 (1996) - 1.1 (1997)- 1.2 (Java2) (1998) - 1.3 (2000) -1.4 (2002) - 1.5 (Java5.0) (2004) - Java 6 (2006) - Java 7 (2011) - Java 8 (Marzo-2014)

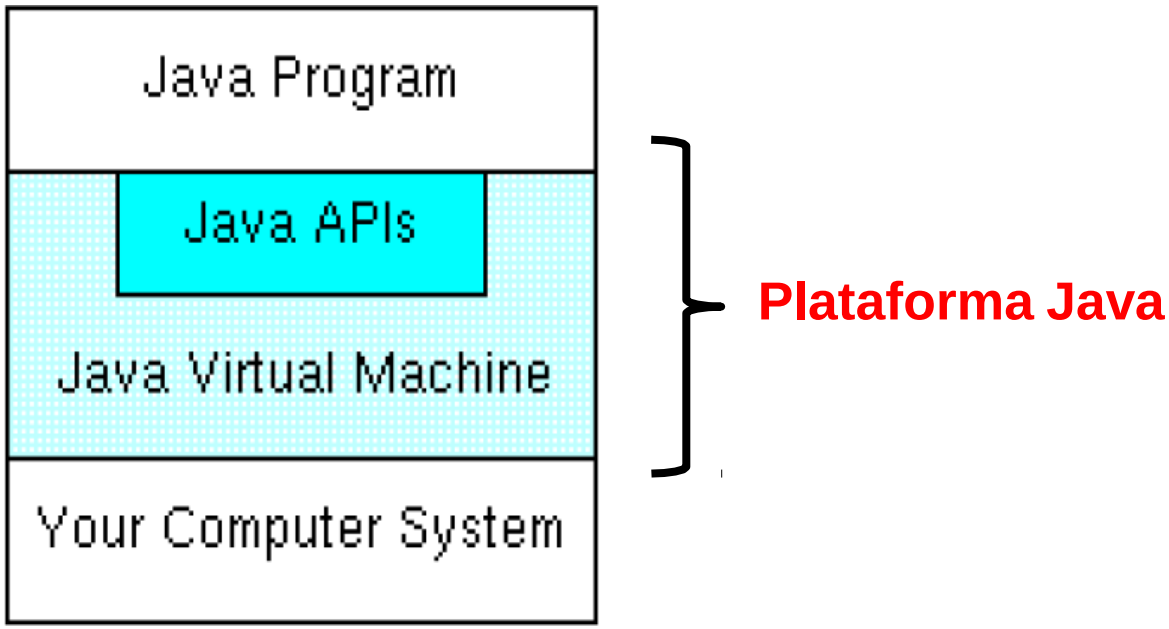
- Java 9 – 2017. Java 10 – 2018. Java 11 - 2018.
  Java 12 – 2019. Java 15 – 2020. Java 16 – 2021
  Java 17 – 2021. Java 18 – 2022
  Java 19 – 2022. Java SE 20 – marzo de 2023

- Múltiples Especificaciones:
  - J8ME (Java 8 Micro Edition)
  - J8SE (Java 8 Standard Edition)
  - J8EE (Java 8 Enterprise Edition)

Plataforma Java

La plataforma Java tiene dos componentes:

- La *Java Virtual Machine (Java VM)*
- La *Interfaz de Programación de Aplicaciones Java (Java API)*



Como muestra la figura, Java API y la máquina virtual (virtual machine) aíslan al programa del hardware

JDK (Java development kit)

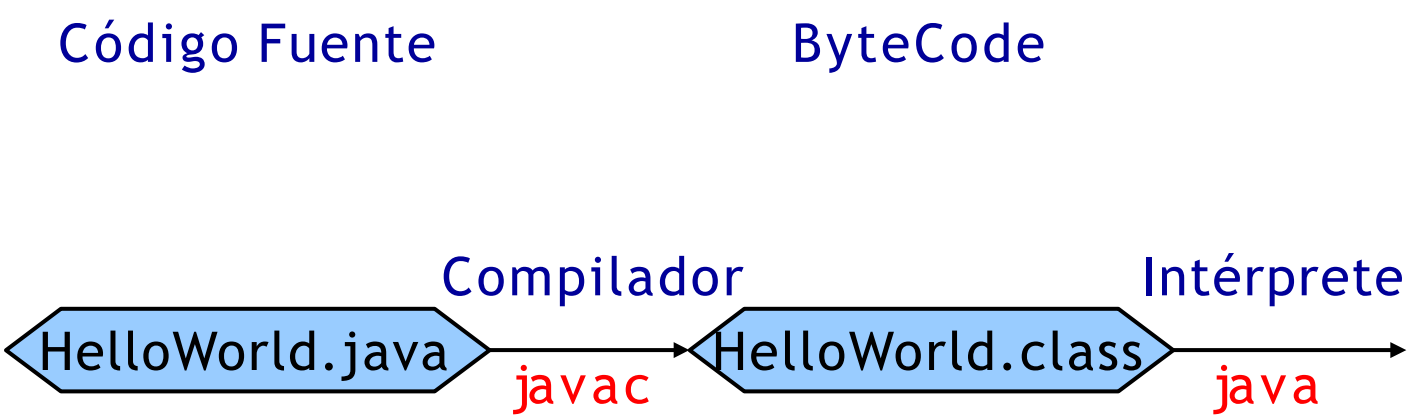
- Compilador: javac
- Intérprete: java
- Plataforma de ejecución: JRE (Java Runtime Environment):
  - Incluye JVM
- Plataforma de desarrollo: Java JDK (Java Software Development Kit):
  - Incluye Compilador, etc.
  - Incluye JRE

Entornos RAD

- Productividad
- Modelado visual
- Depuración
- Rapidez de desarrollo
- Eclipse, Netbeans, IntelliJ IDEA, Visual Code, …

- Prácticas:
  - J8SE (Java8 Standard Edition)
    - Gratuito: http://www.java.com/download
  - Eclipse
    - Gratuito: http://www.eclipse.org
    - Versiones para Windows, Linux, etc.

Ciclo de ejecución



Agenda

- ¿Qué es programar?
- Arquitectura básica de un ordenador
- Breve introducción histórica a la programación
- Compilación vs. interpretación de programas
- Paradigmas de programación
- Programación orientada a objetos: Java
- **Resumen y Referencias**

Resumen

- ¿Qué es programar?
  - Resolver problemas utilizando ordenadores
  - Para resolver un problema se utiliza un algoritmo
  - El algoritmo se escribe en un lenguaje de programación
- Arquitectura básica de un ordenador
  - CPU
  - Memoria
  - Dispositivos de E/S

- Breve introducción histórica a la programación
  - Código Máquina
  - Lenguajes de Bajo Nivel
  - Lenguajes de Alto Nivel
- Compilación vs. interpretación de programas
- Paradigmas de programación
  - Programación Imperativa
  - Programación Funcional
  - Programación Lógica
- Programación Orientada a Objetos: Java
  - Qué es Java - Historia
  - Características principales

Bibliografía y referencias web

- F. Duran y otros.”Programación orientada a objetos con Java” Ed. Thomson, 2007.
- Estadísticas uso lenguajes:
  - http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
- Introducción a la programación:
  - http://es.wikipedia.org/wiki/Lenguaje_de_programación
  - http://manuelpereiragonzalez.blogspot.com/2009/09/historia-de-la-informatica-los.html
  - http://nayar.uan.mx/~iavalos/introprog.htm
  - http://elvex.ugr.es/decsai/java/pdf/2B-Java.pdf
- Programming Intro
  - http://www.landofcode.com/programming-intro/
  - http://www.bfoit.org/itp/
  - http://chortle.ccsu.edu/java5/index.html