



# Módulo 1: Programación en Java

Roger F. González Camacho

Agosto 2023

# Ficha Técnica

## Objetivos

Dar al alumno/a una amplia formación en Java aplicada a las tareas propias de un informático en las empresas, capacitándolo/a para realizar tales tareas mediante el ordenador con eficacia y con el nivel de destreza que requiere el ámbito laboral, utilizando para ello software propietario como libre.

# Ficha Técnica

## Metodología

Para cada tema se explicará la parte teórica y se utilizarán casos prácticos y reales. Luego, el alumno realizará una serie de ejercicios relacionados con cada tema.

**Fechas:** del 01/08/2023 hasta el 14/08/2023

**Nº Horas:** 60

**Horario:** lunes a viernes de 15:30 – 21:30

# Evaluaciones

- Exposición.
- Evaluaciones continuas durante la semana. (Actividades, pruebas, exposiciones e Informes)
- Prueba teórico
- Prueba práctico
- Recuperatorio
- Ponderación:
  - 30% - Evaluaciones continuas.
  - 70% Pruebas Finales
    - 30% Teórico
    - 40% Práctico

# Introducción

Este primer módulo pretende ser una rápida introducción a la programación en Java. En primer lugar muestra lo que es Java, sus características y las herramientas que están ligadas a él y, a continuación, enseña cómo compilar y ejecutar algunos programas sencillos escritos en Java. La tecnología Java es tanto una plataforma como un lenguaje de programación. En los próximos capítulos posteriores se trata de dar una visión más detallada de la sintaxis del lenguaje de programación Java.

# Objetivos

- Describir las características del lenguaje de programación Java.
- Describir las herramientas ligadas a la construcción y ejecución de programas escritos en Java.
- Construir las primeras aplicaciones en Java.

# El Lenguaje de Programación Java

El lenguaje de programación Java, fue diseñado por la compañía Sun Microsystems Inc, con el propósito de crear un lenguaje que pudiera funcionar en sistemas de ordenadores heterogéneos (redes de computadoras formadas por más de un tipo de ordenador, ya sean PC compatibles, Macintosh o estaciones de trabajo que empleen diferentes sistemas operativos como Windows, OS/2 o Unix), y que fuera independiente de la plataforma en la que se vaya a ejecutar. Esto significa que un programa de Java puede ejecutarse en cualquier máquina o plataforma.

# El Lenguaje de Programación Java

Su origen se remonta a la creación de un lenguaje de programación para el desarrollo de aplicaciones para electrodomésticos y otros aparatos electrónicos de consumo por parte de una empresa filial de Sun, llamada FirstPerson en 1991. Su creador, James Gosling, lo bautizó como Oak1. Al abandonarse este proyecto, el lenguaje se modificó, al igual que su nombre y se orientó al desarrollo de aplicaciones para la red. En septiembre de 1995 aparece el primer Kit de Desarrollo de Java (JDK). A principios de 1997 se presenta la primera revisión de Java (la versión 1.1) y a finales de 1998 surge la versión 1.2 (Java 2) que introdujo modificaciones bastante significativas. En octubre de 2004 se hace pública la versión Java 1.5 (Java 5) incluyendo innovaciones muy importantes en la plataforma.



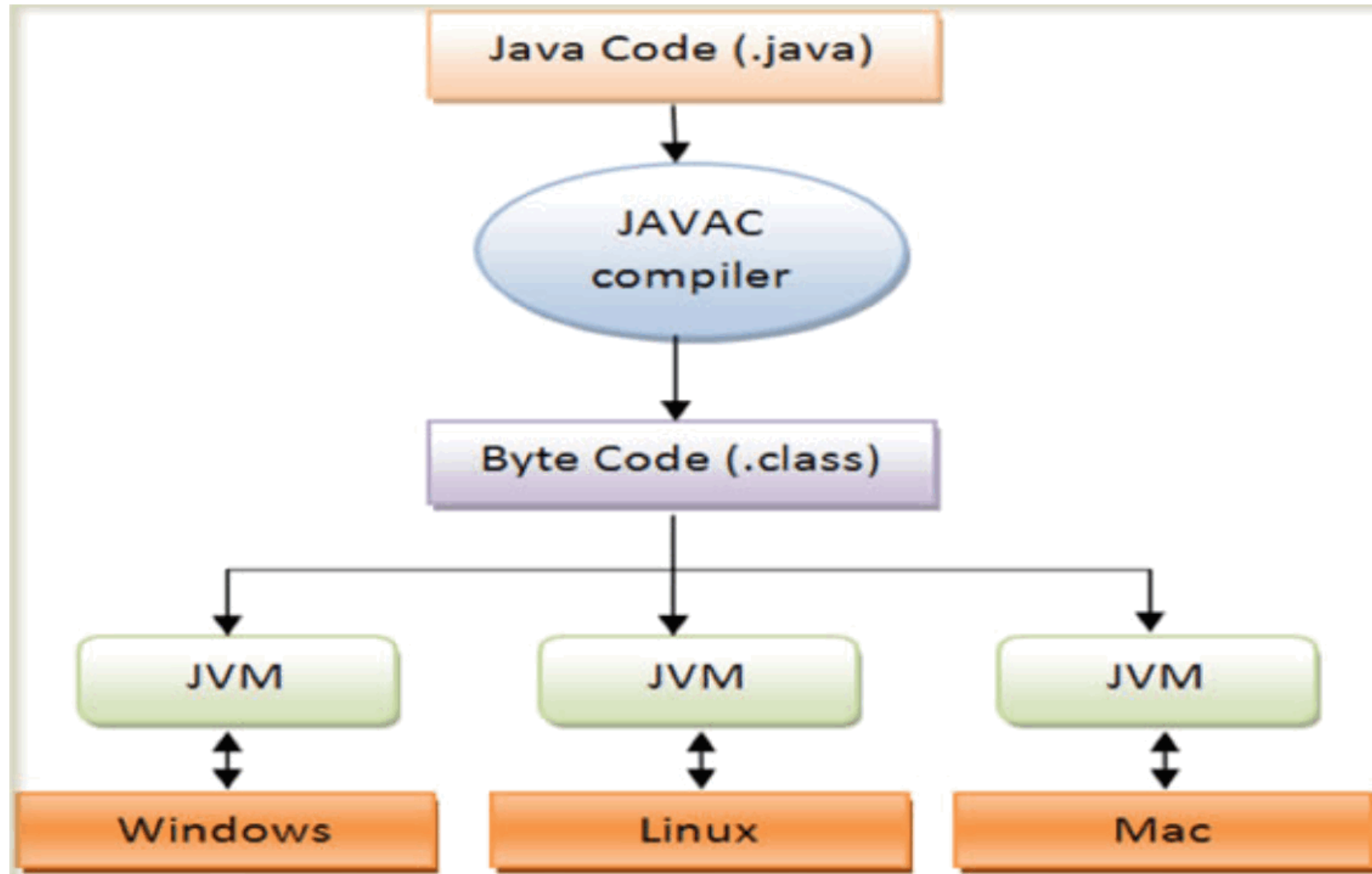
# Histórico de versiones java

Versión	Fecha de Lanzamiento	Comentario
JDK 1.0	enero de 1996	Primera versión
JDK 1.1	febrero de 1997	
J2SE 1.2	diciembre de 1998	Se modificó el nombre bajo la denominación Java 2 y el nombre "J2SE" (Java 2 Platform, Standard Edition), reemplazó a JDK para distinguir la plataforma base de J2EE (Java 2 Platform, Enterprise Edition) y J2ME (Java 2 Platform, Micro Edition).
Java SE 6	diciembre de 2006	En esta versión, Sun cambió el nombre "J2SE" por Java SE.
Java SE 8 (LTS)	marzo de 2014	Nuevas utilidades del lenguaje, destacan las expresiones lambda y las nuevas APIs para manipular fechas y colecciones de objetos.
Java SE 9	septiembre de 2017	
Java SE 12 (19.3)	marzo de 2019	OpenJDK
Java SE 13	septiembre 2019	
Java SE 14	marzo 2020	
<b>Java SE 15</b>	<b>septiembre 2020</b>	
<b>Java SE 16</b>	<b>marzo 2021</b>	
<b>Java SE 17</b>	<b>septiembre 2021</b>	
<b>Java SE 18</b>	<b>marzo 2022</b>	
<b>Java SE 19</b>	<b>septiembre 2022</b>	
<b>Java SE 20</b>	<b>marzo 2023</b>	
<b>Java SE 21 (LTS)</b>	<b>Septiembre 2023</b>	

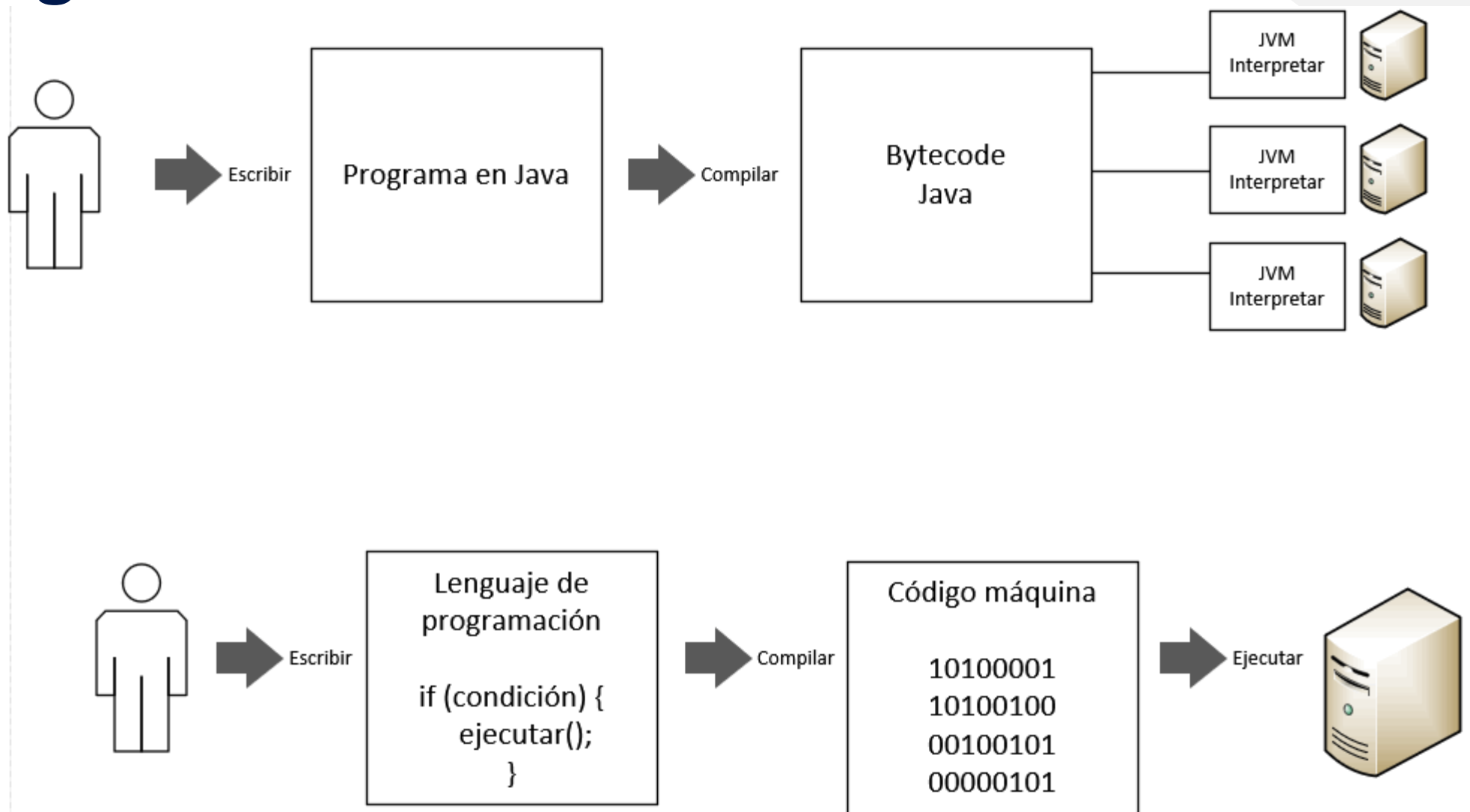
# Características del Lenguaje Java

- **Sencillo:** Elimina la complejidad de los lenguajes como C y da paso al contexto de los lenguajes modernos orientados a objetos.
- **Orientado a Objetos:** La filosofía de programación orientada a objetos es diferente a la programación convencional (imperativa o procedural).
- **Independiente a la arquitectura y portable:** Al compilar un programa en Java, el código resultante es un tipo de código binario conocido como Java bytecodes.
- **Robusto:** Java simplifica la gestión de la memoria dinámica.
- **Seguro:** El sistema de Java tiene ciertas políticas que evitan que se puedan codificar virus con este lenguaje.
- **Multitarea (Multithreaded):** Un lenguaje que soporta múltiples threads, hilos o tareas.

# Mecanismo de Creación de un Programa Java



# Mecanismo de Creación de un Programa Java



# Ventajas en el Uso de Java

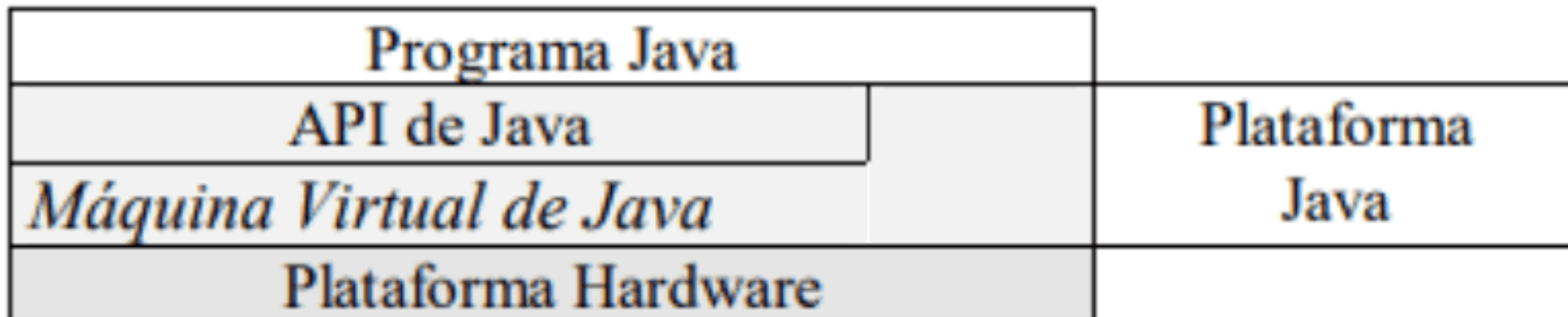
- **Compatibilidad**
- **Funcionalidad:** Si interesa desarrollar un servicio Web con funciones dinámicas más allá de las posibilidades del lenguaje HTML, puede emplearse Java para incluir en las páginas toda clase de elementos multimedia y permitir un alto nivel de interactividad.
- **Ahorro de recursos:** Un navegador compatible con Java deberá ejecutar cualquier programa hecho en Java.
- **Metodología OO**
- **Menos y mejor código**
- **Gratuidad**

# Inconvenientes del Lenguaje Java

- Mayor consumo memoria
- Mayor tiempo de carga de la VM
- Integración no perfecta con el sistema operativo
- Es un lenguaje de programación

# La plataforma Java

- El intérprete, Máquina Virtual Java o Java Virtual Machine (JVM).
- La Interfaz de Programación de Aplicaciones Java o Java Application Programming Interface (Java API).
- El API de Java es una amplia colección de componentes de software que facilitan muchas necesidades de programación como puede ser código necesario para construir una interfaz de usuario (GUI). El API de Java se agrupa en librerías o paquetes (packages) de componentes relacionados entre sí: componentes básicos de programación, creación de Applets, redes, internacionalización, seguridad, componentes de software, conectividad y redes, etcétera. Hay, además, extensiones estándar fuera del núcleo del API de Java que facilitan recursos para servidores, gráficos 3D, animación...



# Herramientas de Desarrollo para Java

## Java SE

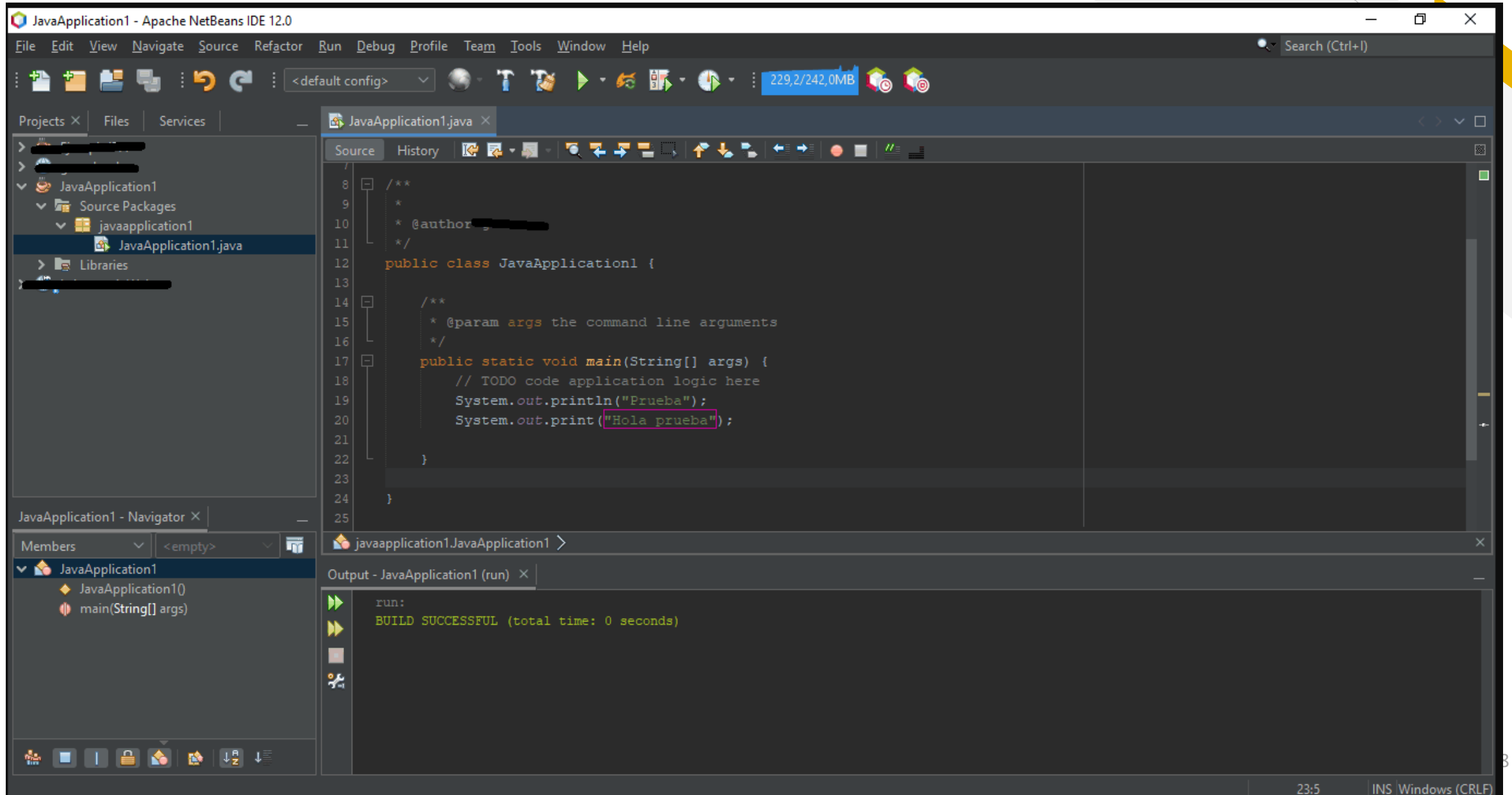
- JRE (Java Runtime Environment). => Usuario final que ejecuta Java en un escritorio.
- Servidor JRE (Server Java Runtime Environment). => Administradores que ejecutan aplicaciones en un servidor
- JDK (Java Development Kit). => Desarrolladores de software, Incluye un JRE completo más herramientas para desarrollar, compilar, depurar y monitorizar aplicaciones Java.



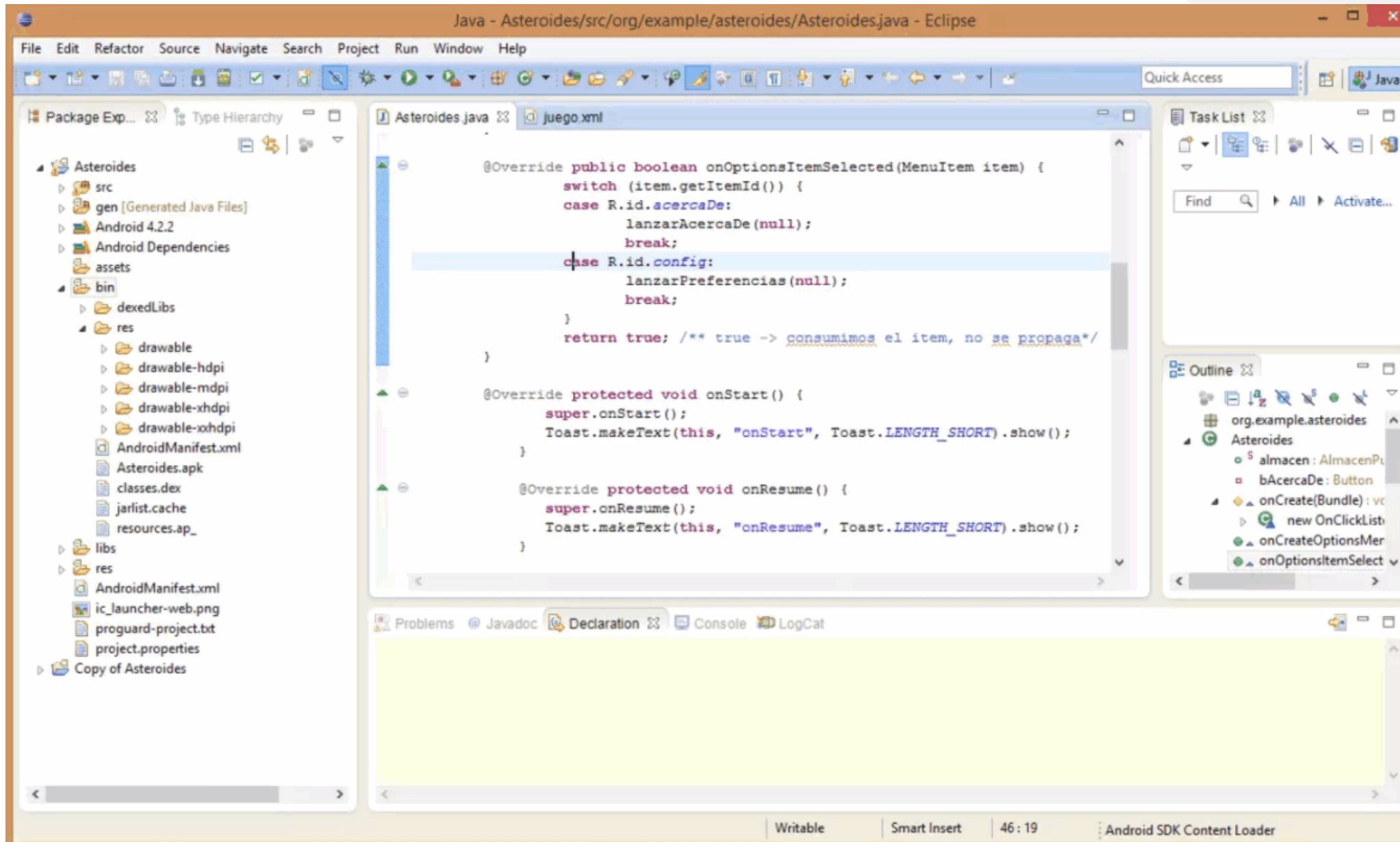
# Entornos de Desarrollo Integrado



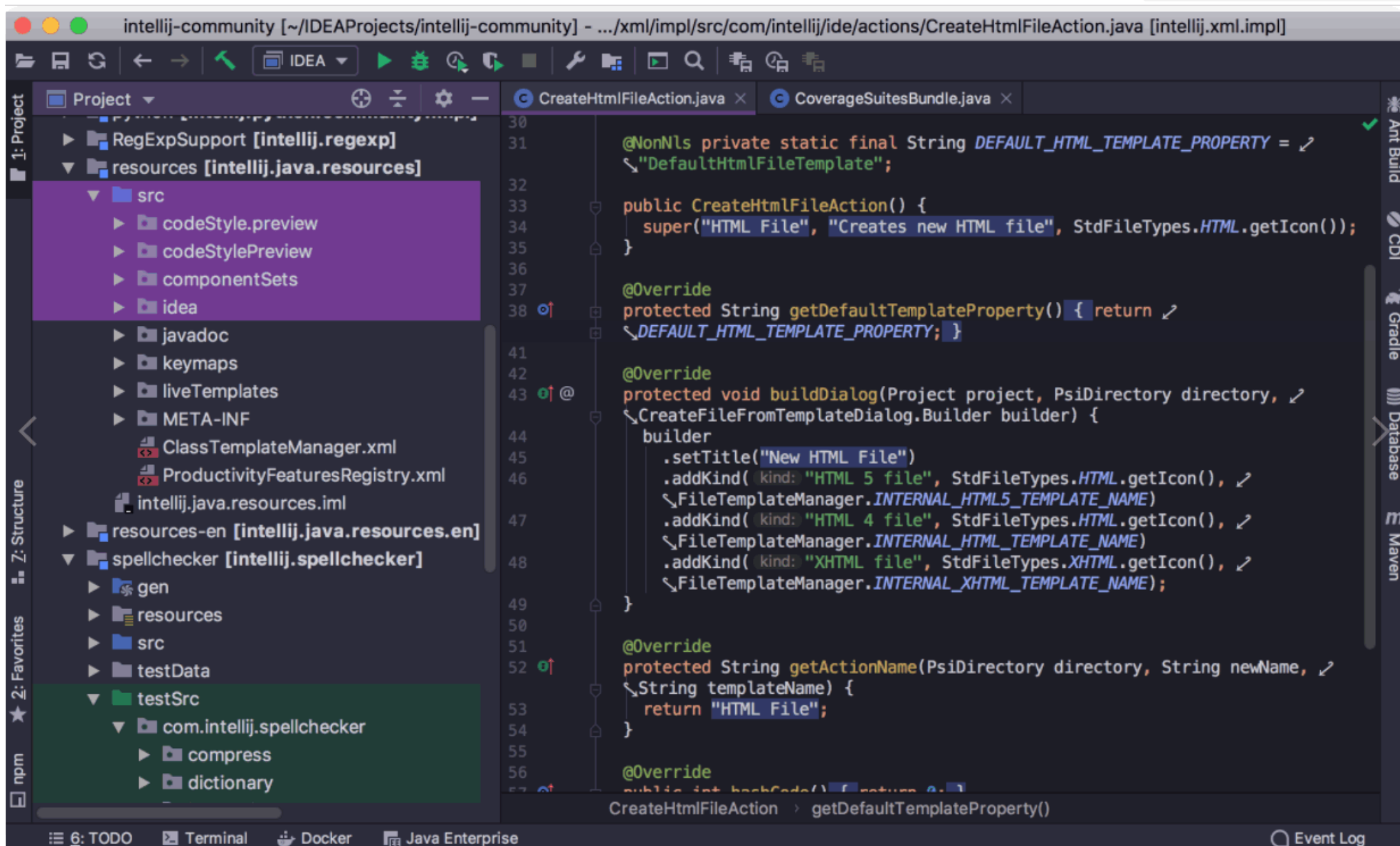
# Netbeans



# Eclipse



# IntelliJ IDEA



# Lenguajes de Programación más Populares



Worldwide, Jul 2023 :

Rank	Change	Language	Share	1-year trend
1		Python	27.43 %	-0.2 %
2		Java	16.19 %	-1.0 %
3		JavaScript	9.4 %	-0.1 %
4		C#	6.77 %	-0.3 %
5		C/C++	6.44 %	+0.2 %
6		PHP	5.03 %	-0.4 %
7		R	4.45 %	+0.1 %
8		TypeScript	3.02 %	+0.3 %
9	↑	Swift	2.42 %	+0.4 %
10	↑↑↑	Rust	2.15 %	+0.6 %
11	↓↓	Objective-C	2.13 %	+0.0 %
12	↓	Go	2.01 %	+0.0 %

# IDE de Populares

# Programación más



Worldwide, Jul 2023 :

Rank	Change	IDE	Share	1-year trend
1		Visual Studio	27.88 %	-0.3 %
2	↑	Visual Studio Code	13.7 %	+1.3 %
3	↓	Eclipse	11.5 %	-1.2 %
4		Android Studio	9.42 %	+0.5 %
5		pyCharm	9.01 %	+0.5 %
6		IntelliJ	7.14 %	+0.2 %
7		NetBeans	4.31 %	-0.7 %
8	↑↑↑	RStudio	3.24 %	+0.5 %
9	↑	Atom	3.19 %	+0.4 %
10	↓↓	Sublime Text	3.1 %	-0.7 %
11	↓↓	Xcode	2.92 %	-0.1 %

# BBDD

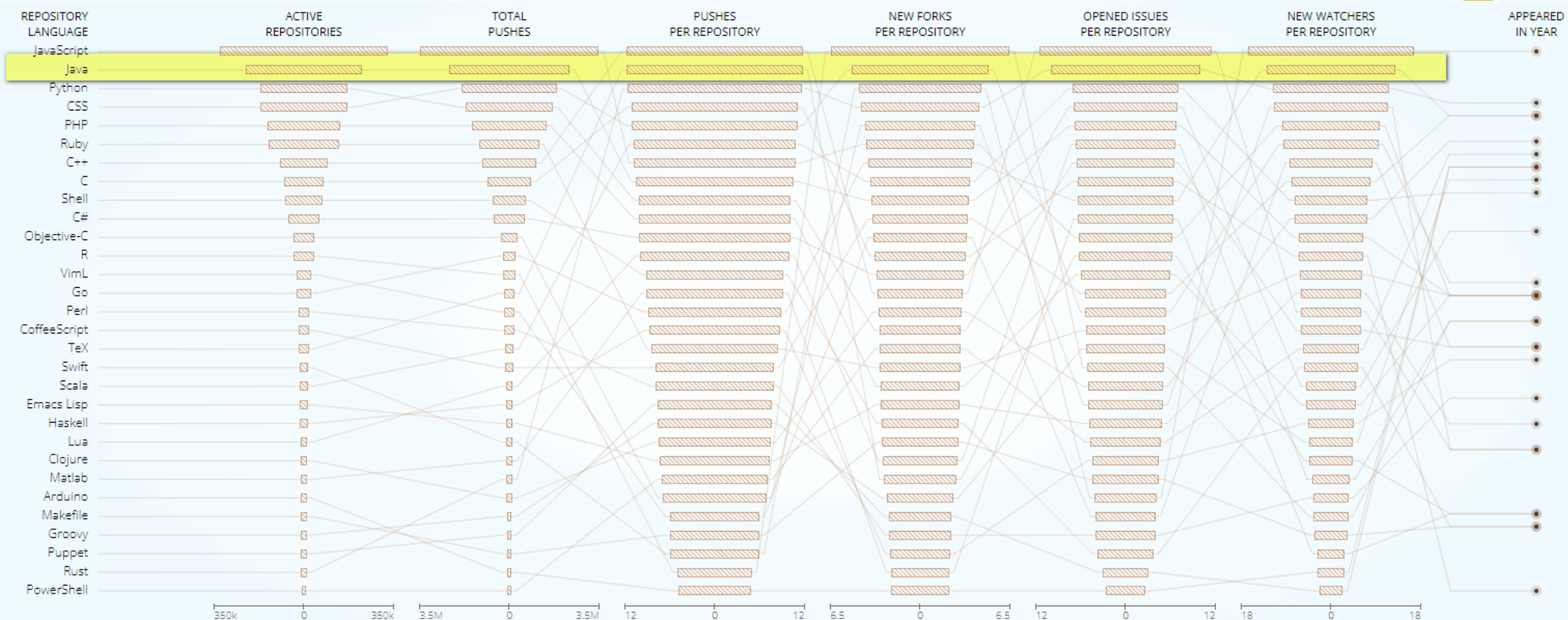
Worldwide, Jul 2023 :

Rank	Change	Database	Share	1-year trend
1		Oracle	27.19 %	-0.1 %
2		MySQL	18.58 %	-0.5 %
3		SQL Server	12.39 %	+0.0 %
4	↑	PostgreSQL	6.74 %	+0.9 %
5	↑	MongoDB	6.12 %	+0.5 %
6	↓↓	Microsoft Access	5.85 %	-1.2 %
7		Firebase	4.81 %	+0.2 %
8		Redis	3.19 %	+0.5 %
9		Splunk	2.54 %	+0.1 %
10		Elasticsearch	2.18 %	-0.1 %
11		SQLite	1.99 %	+0.0 %
12		MariaDB	1.4 %	-0.0 %





# Proyectos GitHub

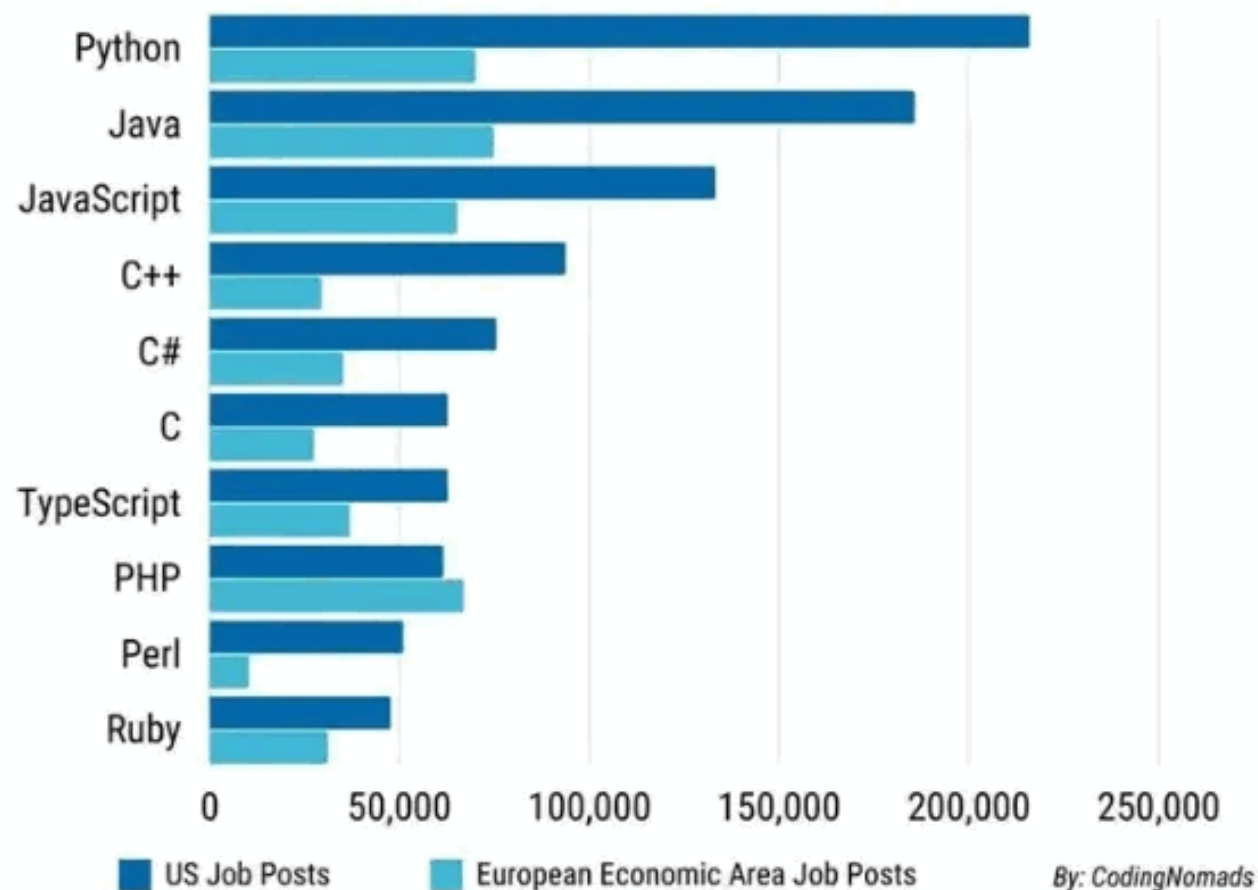




# Lenguajes más demandados

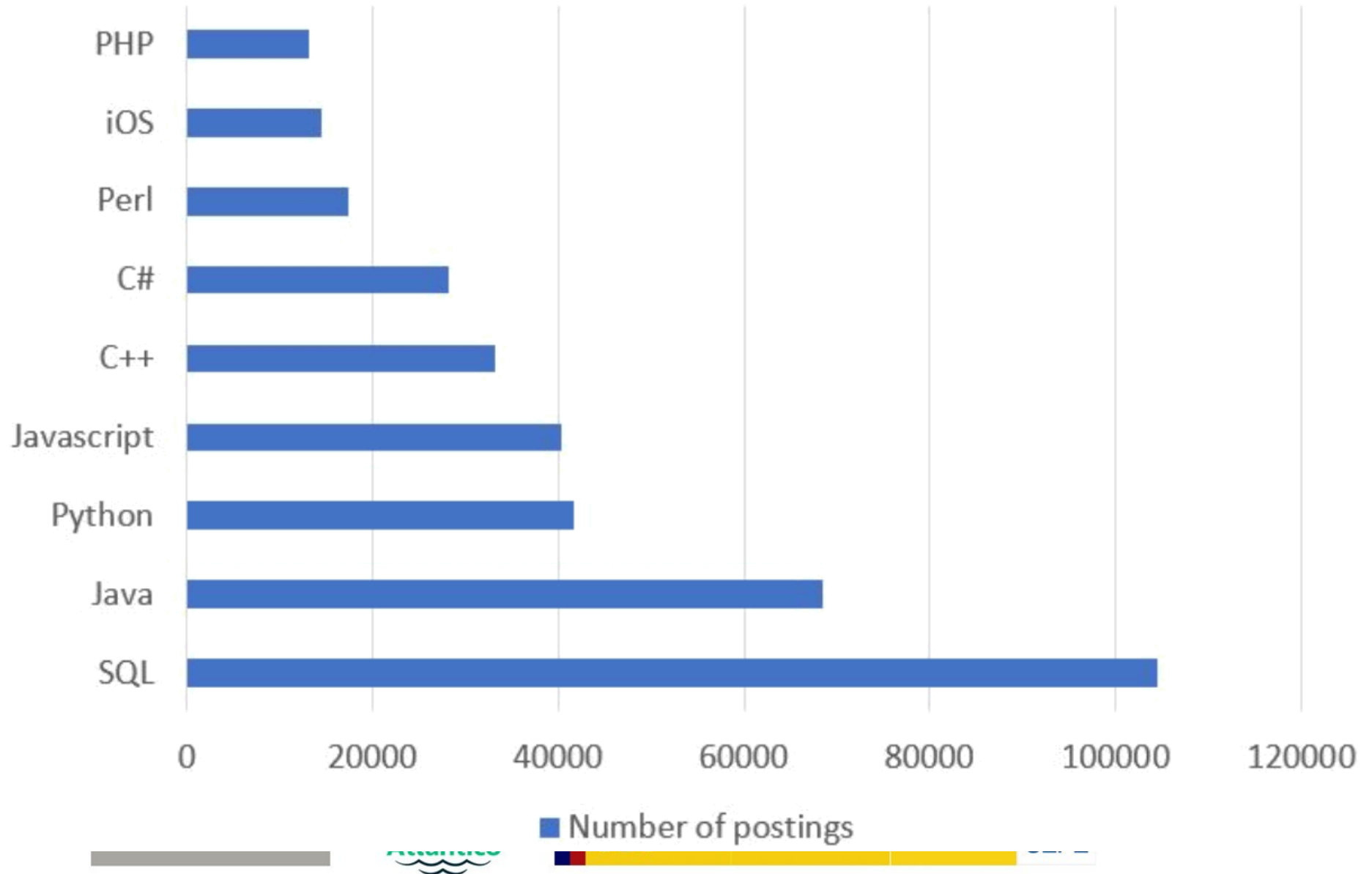
## Most in-demand programming languages of 2022

*Based on LinkedIn job postings in the USA & Europe*



Datos recopilados de la comunidad de LinkedIn en Estados Unidos y Europa.

# Lenguajes más demandados



# Instalación y Configuración de Java

1. Oracle => <https://www.oracle.com/java/technologies/javase/jdk20-archive-downloads.html>
2. OpenJDK
  1. <https://jdk.java.net/>
  2. <https://adoptium.net/es/>
  3. <https://aws.amazon.com/es/corretto/?filtered-posts.sort-by=item.additionalFields.createdDate&filtered-posts.sort-order=desc>
  4. <https://developers.redhat.com/products/openjdk/download>
  5. <https://www.microsoft.com/openjdk>

# Instalación y Configuración de Java

## 1. Descargar las Java Development Kit

<https://www.oracle.com/java/technologies/downloads/#jdk17-windows>

The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.

Documentation Download

Linux macOS Windows

Product/file description	File size	Download
x64 Compressed Archive	170.64 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip</a> (sha256 <a href="#">↗</a> )
x64 Installer	151.99 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe</a> (sha256 <a href="#">↗</a> )
x64 MSI Installer	150.88 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi</a> (sha256 <a href="#">↗</a> )

# Instalación y Configuración de Java

## 1. Descargar las Java Development Kit


<https://www.oracle.com/java/technologies/javase/jdk20-archive-downloads.html>

Linux x64 Debian Package	155.91 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_linux-x64_bin.deb">https://download.oracle.com/java/20/archive/jdk-20.0.2_linux-x64_bin.deb</a> (sha256)
Linux x64 RPM Package	182.82 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_linux-x64_bin.rpm">https://download.oracle.com/java/20/archive/jdk-20.0.2_linux-x64_bin.rpm</a> (sha256)
macOS Arm 64 Compressed Archive	177.21 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-aarch64_bin.tar.gz">https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-aarch64_bin.tar.gz</a> (sha256)
macOS Arm 64 DMG Installer	176.54 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-aarch64_bin.dmg">https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-aarch64_bin.dmg</a> (sha256)
macOS x64 Compressed Archive	179.54 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-x64_bin.tar.gz">https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-x64_bin.tar.gz</a> (sha256)
macOS x64 DMG Installer	178.87 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-x64_bin.dmg">https://download.oracle.com/java/20/archive/jdk-20.0.2_macos-x64_bin.dmg</a> (sha256)
Windows x64 Compressed Archive	180.99 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.zip">https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.zip</a> (sha256)
Windows x64 Installer	160.12 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.exe">https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.exe</a> (sha256)
Windows x64 msi Installer	158.90 MB	<a href="https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.msi">https://download.oracle.com/java/20/archive/jdk-20.0.2_windows-x64_bin.msi</a> (sha256)

# Instalación y Configuración de Java

1. Descargar las Java Development Kit


<https://www.oracle.com/java/technologies/javase-jdk15-downloads.html>



You must accept the [Oracle Technology Network License Agreement for Oracle Java SE](#) to download this software.

---

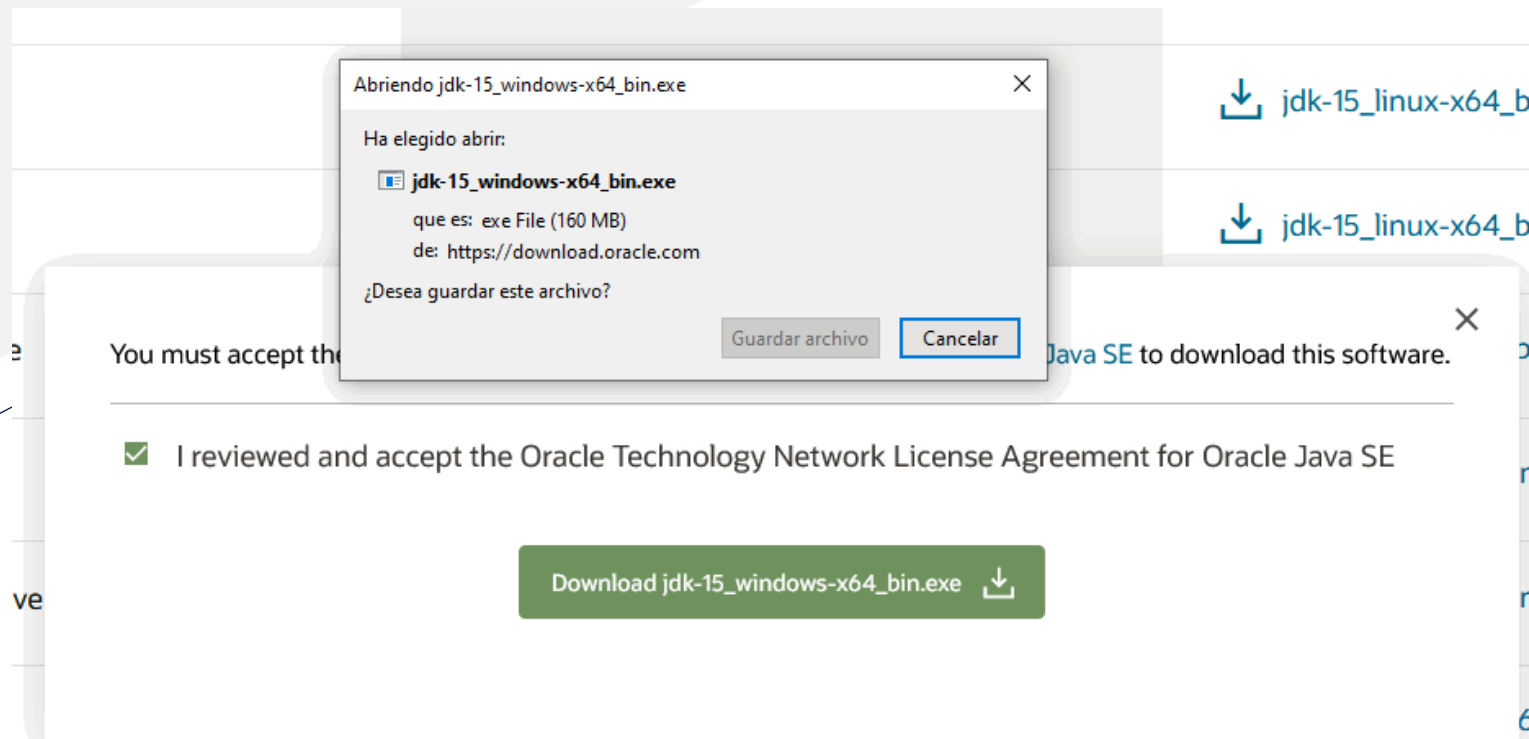
☒ I reviewed and accept the Oracle Technology Network License Agreement for Oracle Java SE

Download jdk-15\_windows-x64\_bin.exe 

# Instalación y Configuración de Java

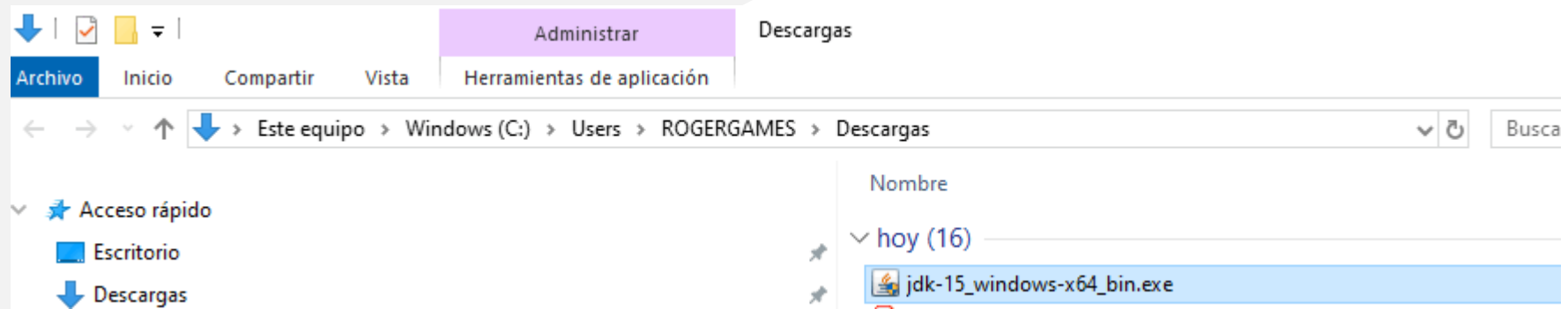
1. Descargar las Java Development Kit

<https://www.oracle.com/java/technologies/javase-jdk15-downloads.html>



# Instalación y Configuración de Java

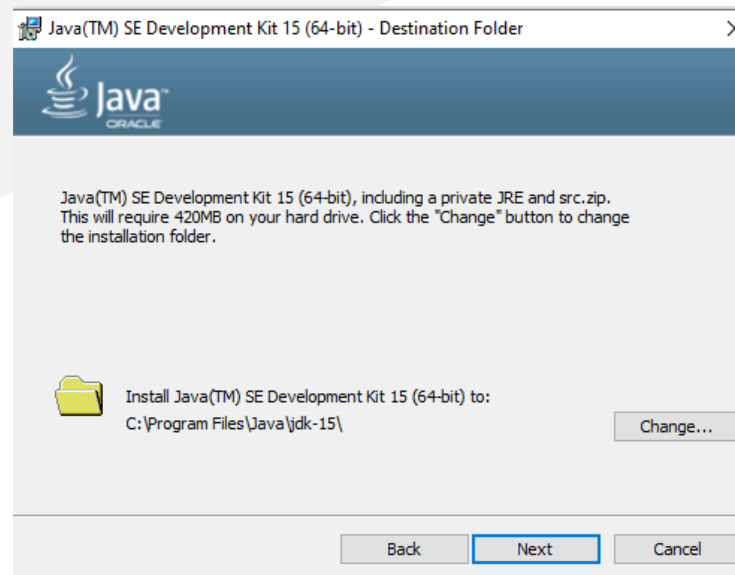
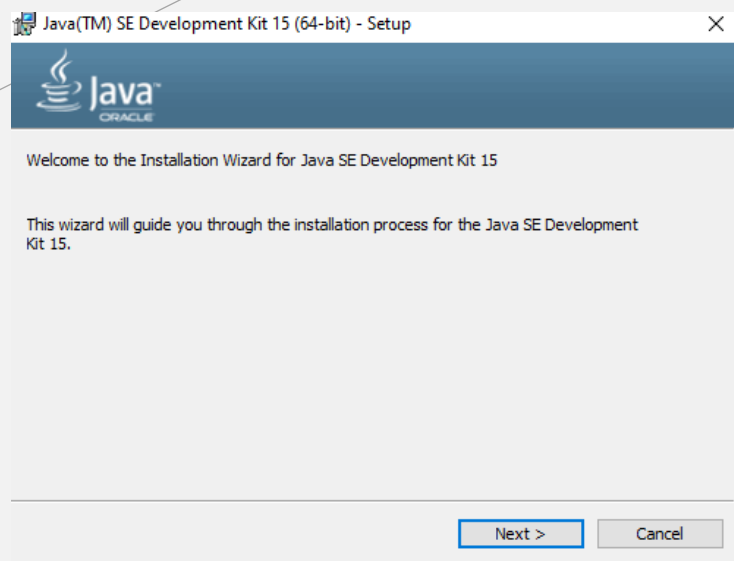
## 2. Ejecutar e instalar el JDK





# Instalación y Configuración de Java

## 2. Ejecutar e instalar el JDK



# Instalación y Configuración de Java

## 2. Ejecutar e instalar el JDK



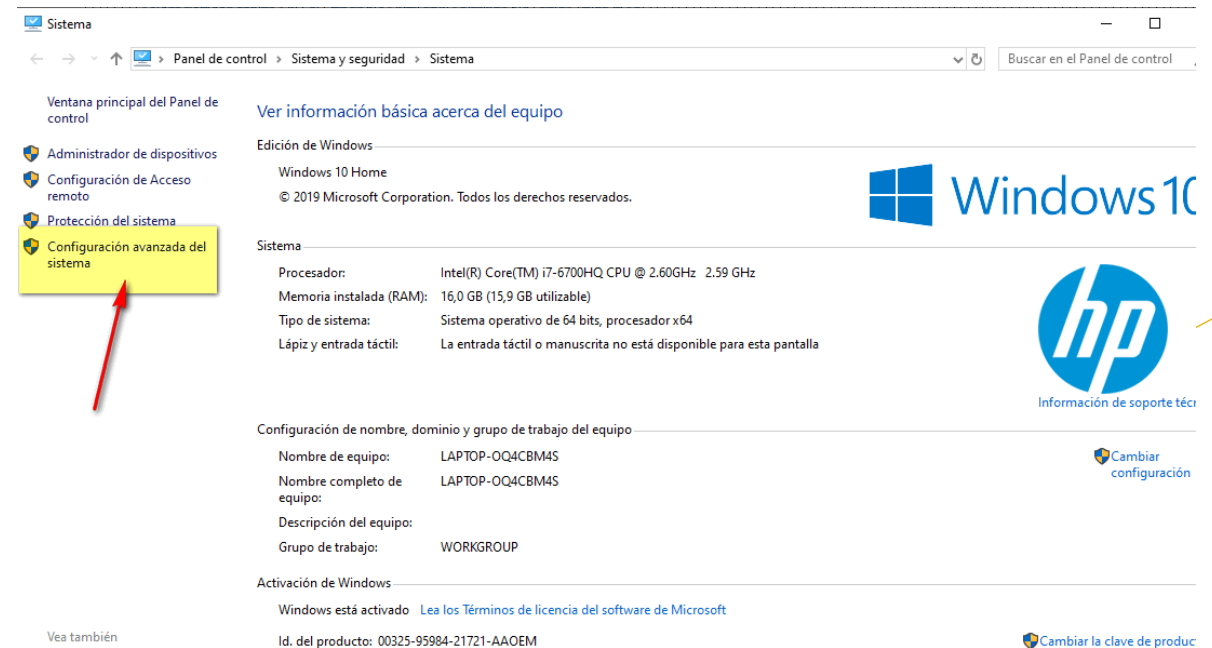
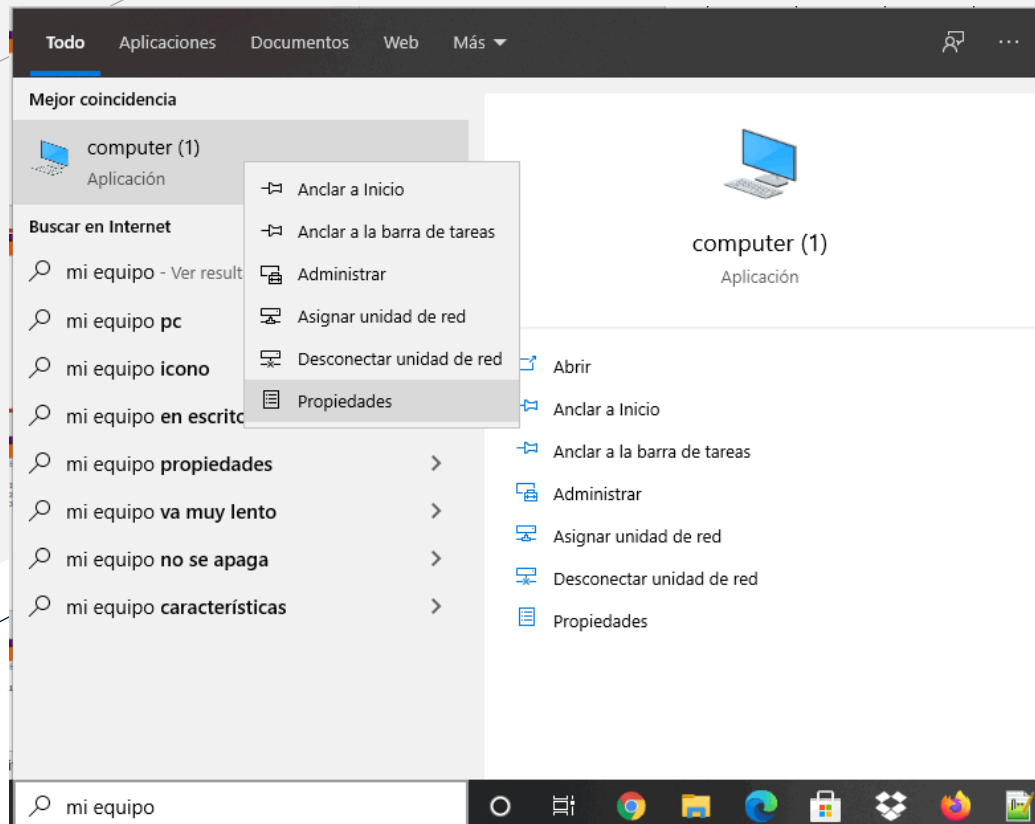
## Instalación y Configuración de Java

### 3. Configurar la variable del sistema PATH

PATH es la variable del sistema que utiliza el sistema operativo para buscar los ejecutables necesarios desde la línea de comandos o la ventana Terminal. Debemos añadir a la variable PATH la ruta donde se encuentran los ejecutables `java.exe` y `javac.exe`.

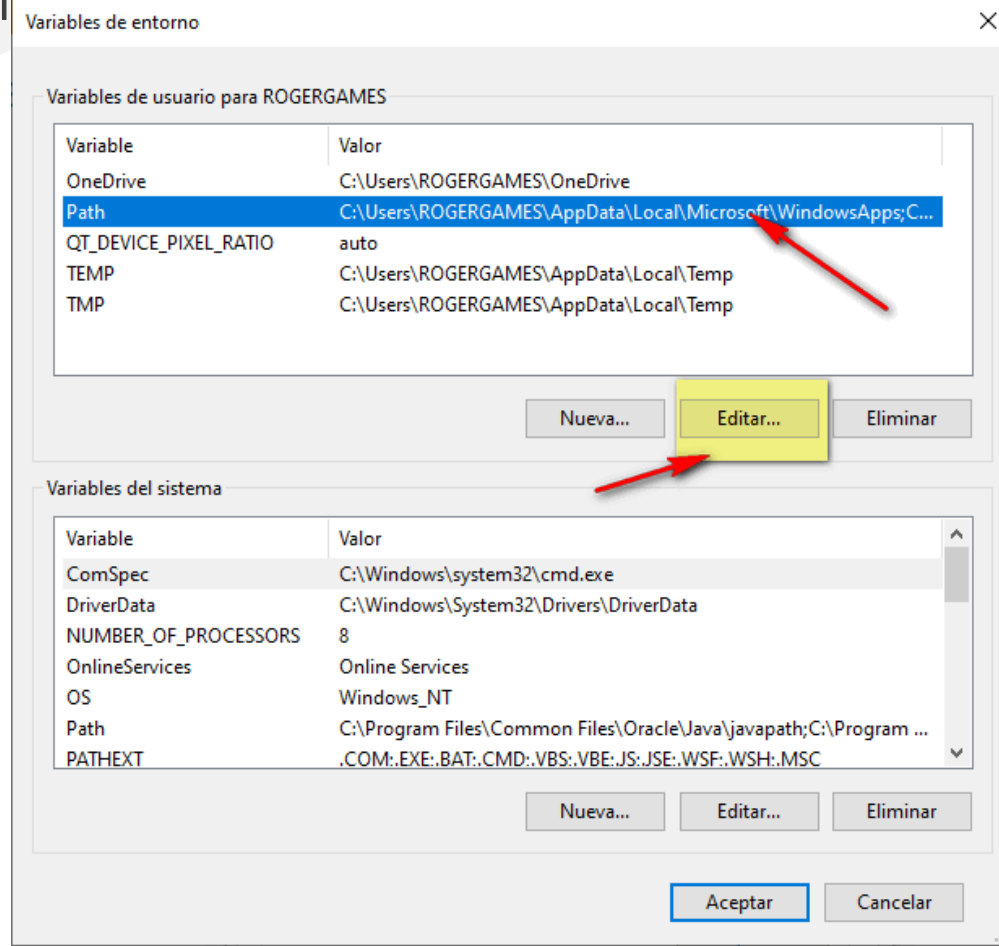
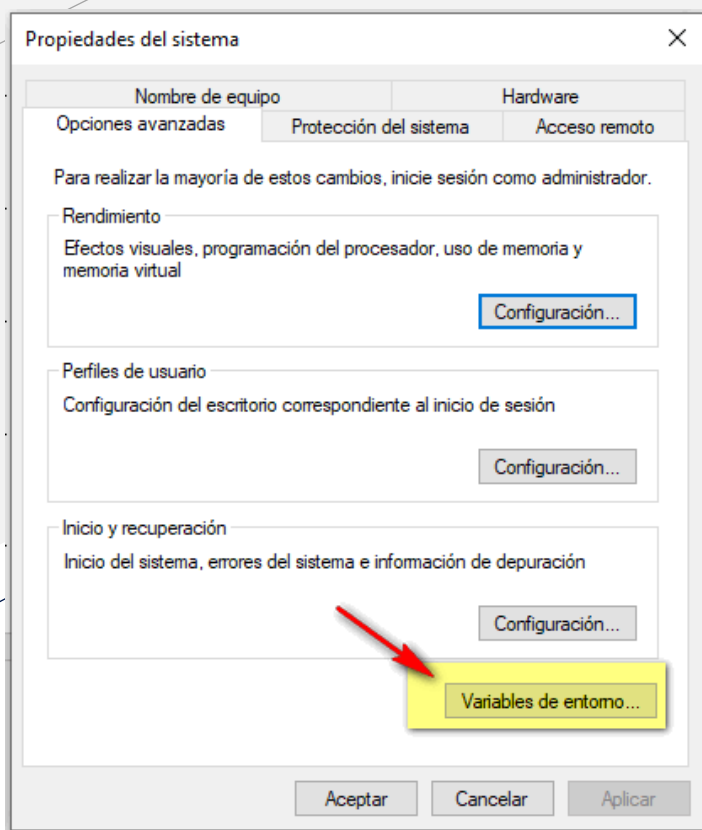
# Instalación y Configuración de Java

## 3. Configurar la variable del sistema PATH



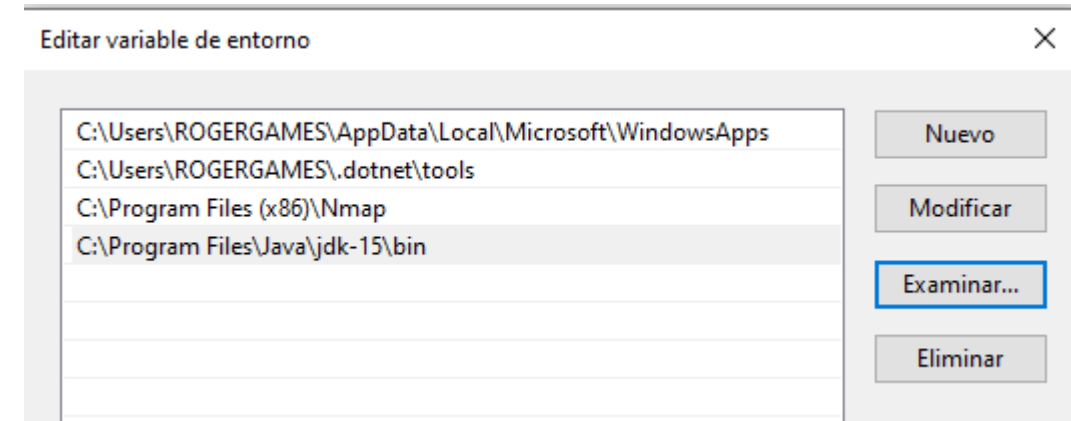
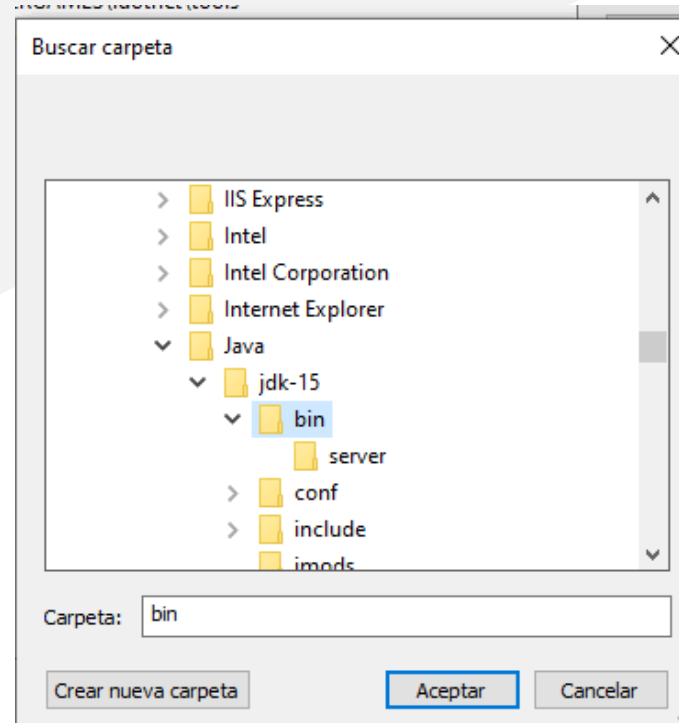
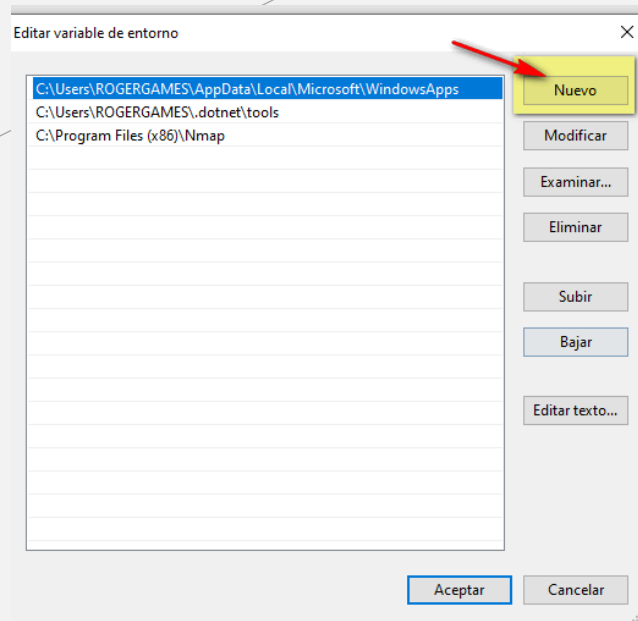
# Instalación y Configuración de Java

## 3. Configurar la variable del sistema PATH



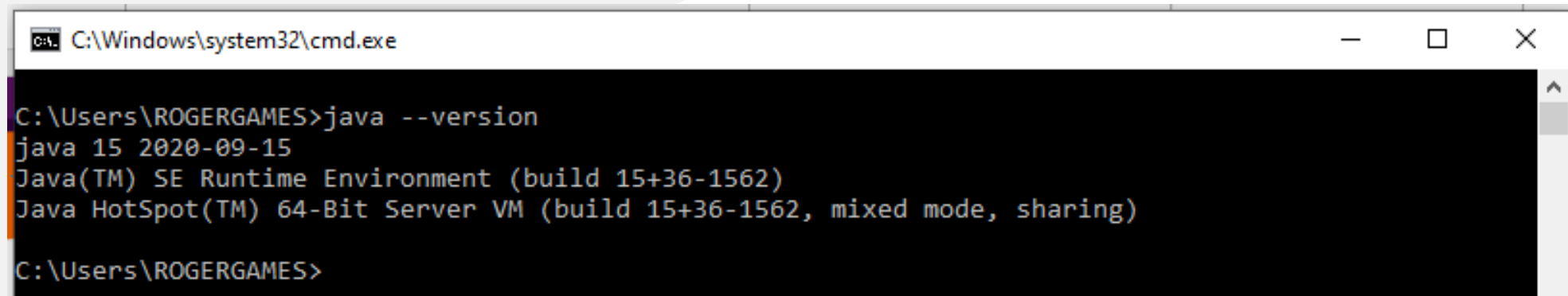
# Instalación y Configuración de Java

## 3. Configurar la variable del sistema PATH



# Instalación y Configuración de Java

## 3. Configurar la variable del sistema PATH



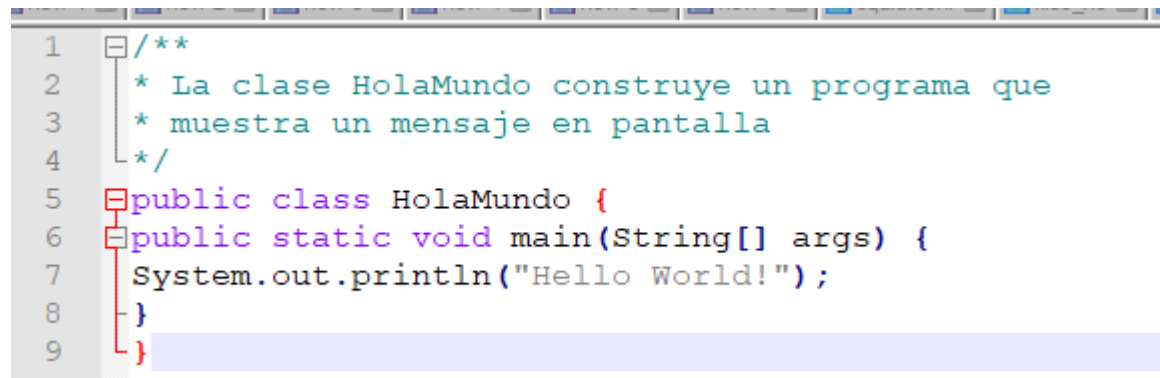
```
C:\Windows\system32\cmd.exe

C:\Users\ROGERGAMES>java --version
java 15 2020-09-15
Java(TM) SE Runtime Environment (build 15+36-1562)
Java HotSpot(TM) 64-Bit Server VM (build 15+36-1562, mixed mode, sharing)

C:\Users\ROGERGAMES>
```

# Mi Primer HolaMundo

```
/**
 * La clase HolaMundo construye un programa que
 * muestra un mensaje en pantalla
 */
class HolaMundo {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```



```
1  /**
2   * La clase HolaMundo construye un programa que
3   * muestra un mensaje en pantalla
4   */
5  public class HolaMundo {
6      public static void main(String[] args) {
7          System.out.println("Hello World!");
8      }
9  }
```



# Mi Primer HolaMundo

```
C:\Windows\system32\cmd.exe

C:\FuenteJava>type HolaMundo.java
/**
 * La clase HolaMundo construye un programa que
 * muestra un mensaje en pantalla
 */
class HolaMundo {
public static void main(String[] args) {
System.out.println("Hello World!");
}
}

C:\FuenteJava>javac HolaMundo.java

C:\FuenteJava>dir/w
El volumen de la unidad C es Windows
El número de serie del volumen es: A0D6-504C

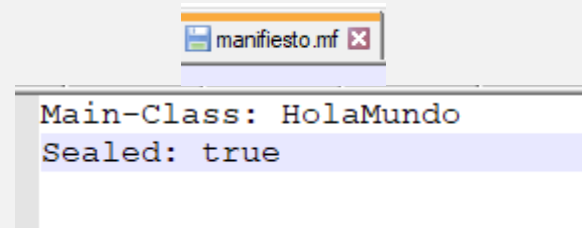
Directorio de C:\FuenteJava

[.]                [..]                HolaMundo.class    HolaMundo.java
                  2 archivos                617 bytes
                  2 dirs  95.840.317.440 bytes libres

C:\FuenteJava>java HolaMundo.java
Hello World!

C:\FuenteJava>
```

# Mi Primer HolaMundo



Este equipo > Windows (C:) > FuenteJava

Nombre	Fecha de modificación	Tipo	Tamaño
HolaMundo.class	07/10/2020 15:49	Archivo CLASS	1 KB
HolaMundo.jar	14/10/2020 12:14	Executable Jar File	1 KB
HolaMundo.java	07/10/2020 15:47	Archivo JAVA	1 KB
manifiesto.mf	14/10/2020 12:13	Archivo MF	1 KB
temp.mf	14/10/2020 11:59	Archivo MF	1 KB

```
C:\FuenteJava>dir/w
```

```
El volumen de la unidad C es Windows
```

```
El número de serie del volumen es: A0D6-504C
```

```
Directorio de C:\FuenteJava
```

```
[.]                [..]                HolaMundo.class  HolaMundo.java  manifiesto.mf  temp.mf
                  4 archivos                675 bytes
                  2 dirs  66.219.880.448 bytes libres
```

```
C:\FuenteJava>jar -cf HolaMundo.jar HolaMundo.class
```

```
C:\FuenteJava>jar -cfm HolaMundo.jar manifiesto.mf HolaMundo.class
```

```
C:\FuenteJava>java -jar HolaMundo.jar
Hello World!
```

```
C:\FuenteJava>
```

# Actividad 1

Redactar un trabajo en el que se hable de un lenguaje de programación distinto a Java. Puedes elegir cualquier lenguaje de programación que esté en índice TIOBE por ejemplo (<https://www.tiobe.com/tiobe-index>). El trabajo debe tener al menos los siguientes apartados:

1. **Introducción al lenguaje.**
2. **Historial de versiones.**
3. **Versión actual.**
4. **Posición en índice TIOBE, índice PYPL y GitHub .**
5. **Entorno de Desarrollo**
  - a. **Ejecutable**
  - b. **Editores de Texto o Entornos de Desarrollo Integrado.**
6. **Ejemplo de código**
7. **Aplicaciones o empresas que usan esta tecnología**

El trabajo no debe **tener más de seis páginas** y se tendrá que exponer en clase.

# Nociones básicas de programación

Generalmente, la primera razón que mueve a una persona hacia el aprendizaje de la programación es utilizar el ordenador como herramienta para resolver problemas concretos. Como en la vida real, la búsqueda y obtención de una solución a un problema determinado, utilizando medios informáticos, se lleva a cabo siguiendo unos pasos fundamentales.

<i>Resolución de</i>	
En la vida real...	En
<b>Observación de la situación o problema.</b>	<b>Análisis del problema:</b> requiere que el problema sea definido y comprendido claramente para que pueda ser analizado con todo detalle.
<b>Pensamos en una o varias posibles soluciones.</b>	<b>Diseño o desarrollo de algoritmos:</b> procedimiento paso a paso para solucionar el problema dado.
<b>Aplicamos la solución que estimamos más adecuada.</b>	<b>Resolución del algoritmo elegido en la computadora:</b> consiste en convertir el algoritmo en programa, ejecutarlo y comprobar que soluciona verdaderamente el problema.

# ¿Qué es un programa?

Conjunto de instrucciones que entiende un ordenador para realizar una actividad. Todo programa tiene un objetivo bien definido: un procesador de texto es un programa que permite cargar, modificar e imprimir textos, un programa de ajedrez permite jugar al ajedrez contra el ordenador u otro contrincante humano.

La actividad fundamental del programador es resolver problemas empleando el ordenador como herramienta fundamental.

Para la resolución de un problema hay que plantear un algoritmo. Algoritmo: Son los pasos a seguir para resolver un problema.

# Representación gráfica

Para representar gráficamente los algoritmos que vamos a diseñar, tenemos a nuestra disposición diferentes herramientas que ayudarán a describir su comportamiento de una forma precisa y genérica, para luego poder codificarlos con el lenguaje que nos interese. Entre otras tenemos:

**Pseudocódigo:** Esta técnica se basa en el uso de palabras clave en lenguaje natural, constantes (Estructura de datos que se utiliza en los lenguajes de programación que no puede cambiar su contenido en el transcurso del programa.), variables (Estructura de datos que, como su nombre indica, puede cambiar de contenido a lo largo de la ejecución de un programa.), otros objetos, instrucciones y estructuras de programación que expresan de forma escrita la solución del problema. Es la técnica más utilizada actualmente.

# Pseudocódigo

Dado que no existe una sintaxis estandarizada para la escritura de pseudocódigo, es posible encontrar diferencias sustanciales por diferente programadores.....

*Programa: Radio del círculo*

**Entorno:**

*constante double PI = 3.1416*

*entero radio;*

*double area;*

**Inicio:**

*Mostrar "Introduce el radio: ";*

*Leer radio;*

*area = pi \* radio2*

*Mostrar "El área del círculo es: " area;*

**Fin**

# Operadores

Aritméticos		Relacionales		Lógicos	
		(Usados para formar condiciones)		(Usados para formar condiciones)	
+	Suma	=	Igual	and	y lógico (conjunción)
-	Resta	<	Menor	or	o lógico (disyunción)
*	Multiplicación	≤	Menor o igual	no	Negación lógica
/	División real	>	Mayor	<b>Especiales</b>	
div	División entera	≥	Mayor o igual	←	Asignación
mod o ÷	Resto o módulo	<>	Distinto	//	Comentario
^	Potencia				



# Palabras reservadas y tipos de datos

Inicio	Si no	Otro	Para	En
Fin	Según	Mientras	Hasta	Procedimiento
Si	Hacer	Repetir	Incremento	Función
Entonces	Caso	Hasta que	Cada	Imprimir
Leer	Retornar			

Carácter	Cadena	Entero	Real	Booleano
----------	--------	--------	------	----------

# Ejemplos

```
Inicio
  <instrucción1>
  ...
  <instrucciónN>
Fin
```

```
Si <condición> entonces
  <instrucción1>
  ...
  <instrucciónX>
Fin Si
```

```
Si <condición> entonces
  <instrucciones1>
Si no
  <instrucciones2>
Fin Si
```

Equivalentes

```
Según <expresión> hacer
  Caso <valor1>
    <instrucciones1>
  Caso <valor2>
    <instrucciones2>
  Caso <valor3>
    <instrucciones3>
  Otro caso
    <instruccionesN>
Fin Según
```

```
Si <condición1> entonces
  <instrucciones1>
Si no
  Si <condición2> entonces
    <instrucciones2>
  Si no
    Si <condición3> entonces
      <instrucciones3>
    Si no
      <instruccionesN>
    Fin Si
  Fin Si
Fin Si
```



# Ejemplos

```
Mientras <condición> Hacer  
    <instrucciones>  
Fin Mientras
```

```
Repetir  
    <instrucciones>  
Hasta Que <condición>
```

```
Hacer  
    <instrucciones>  
Mientras <condición>
```

```
entero i  
Para i ← 1 Hasta N Incremento 1 Hacer  
    <instrucciones>
```

Incremento  
positivo

```
entero i  
Para i ← N Hasta 1 Incremento -1 Hacer  
    <instrucciones>
```

Incremento  
negativo

```
entero i  
Para Cada elemento En conjunto Hacer  
    <instrucciones>  
Fin Para Cada
```

# Ejemplos

```
Procedimiento nombre (tipo parámetro1, tipo parámetro2, ...)  
    <instrucciones>  
Fin Procedimiento
```

```
Procedimiento HolaMundo ()  
    imprimir ("Hola Mundo")  
    imprimir ("-----")  
Fin Procedimiento
```

```
Inicio  
    HolaMundo ()  
Fin
```

```
Procedimiento Saludo (cadena nombre)  
    imprimir ("Hola" + nombre)  
Fin Procedimiento  
Inicio  
    cadena nombre  
    imprimir ("Introduzca su nombre: ")  
    leer (nombre)  
    Saludo (nombre)  
Fin
```



# Ejemplos

```
Función nombre (tipo parámetro1, tipo parámetro2, ...) : tipoRetorno  
    <instrucciones>  
    ...  
    retornar X  
Fin Función
```

# Ejemplos

```
Función TextoParImpar (entero número) : cadena
  cadena resultado
  Si (número % 2 = 0) entonces
    resultado ← "par"
  Si no
    resultado ← "impar"
  Fin Si
  retornar resultado
Fin Función
```

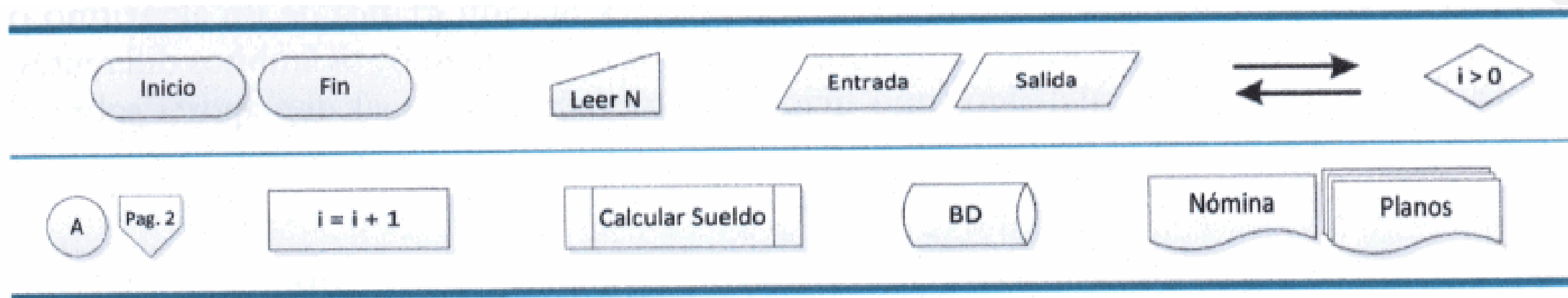
```
Inicio
  entero número
  imprimir ("introduzca el número ...")
  leer (número)
  imprimir ("el número introducido es " + TextoParImpar (número))
Fin
```

Declaración  
de la función

Llamada  
a la función

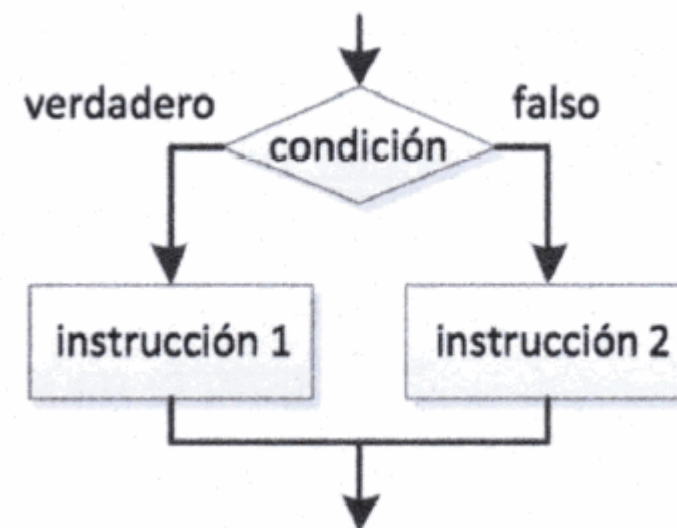
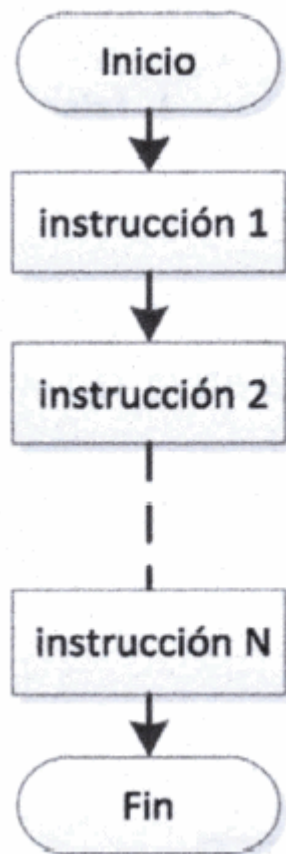
# Diagramas de flujos

Esta técnica utiliza símbolos gráficos para la representación del algoritmo. Suele utilizarse en las fases de análisis.



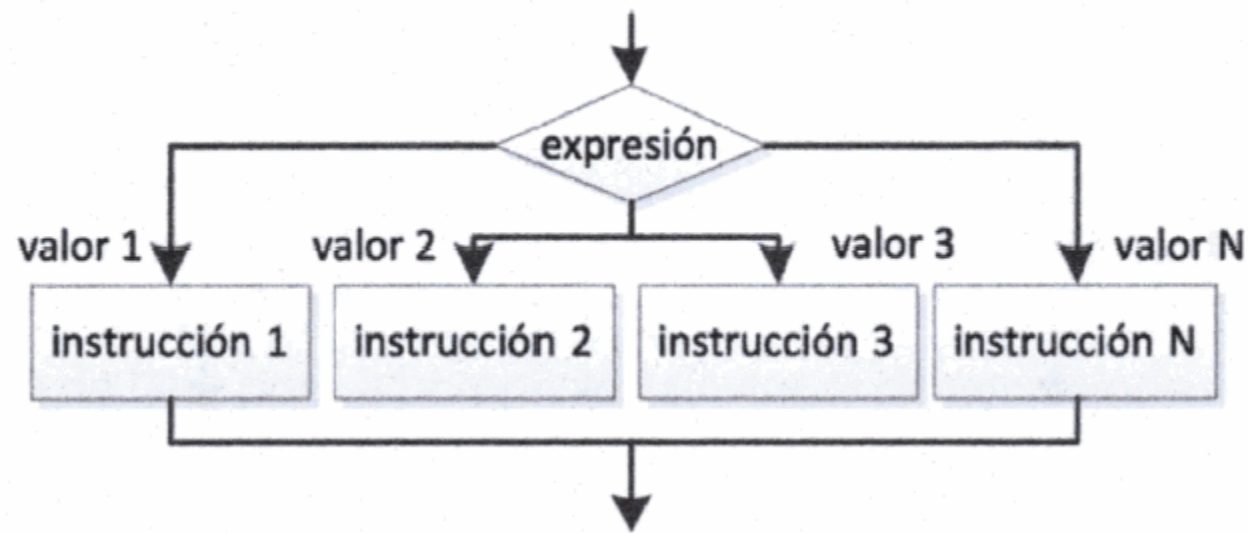
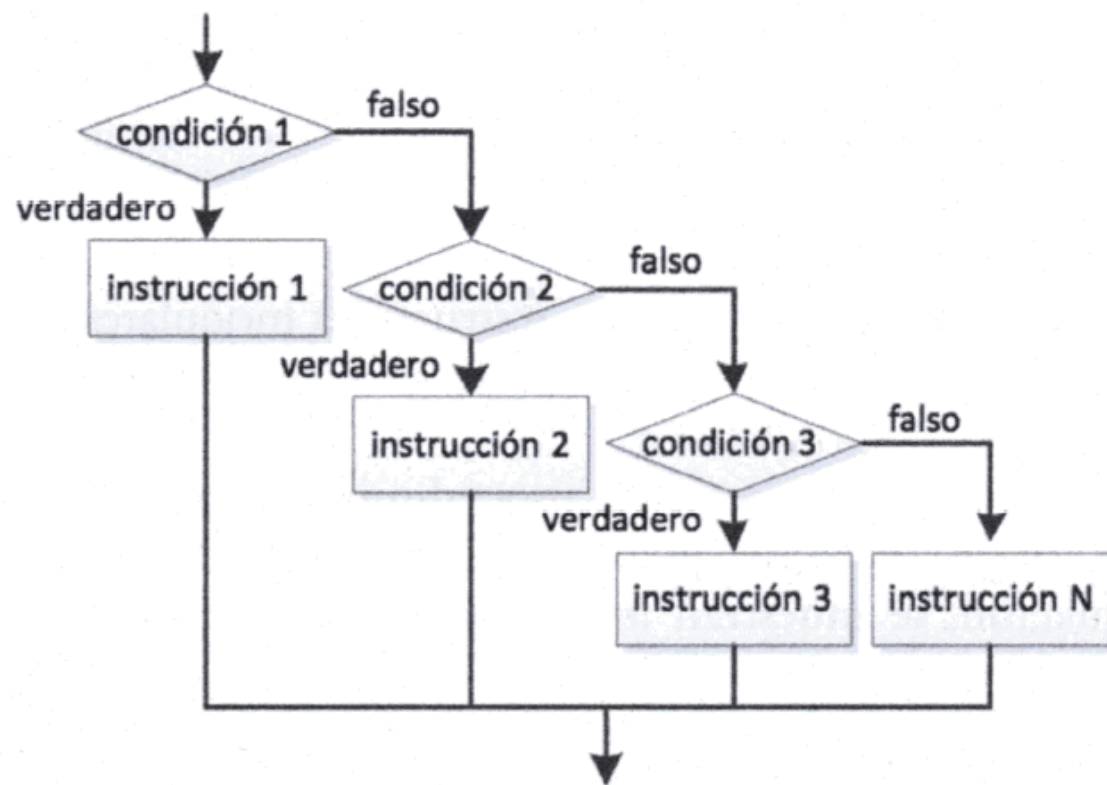


# Ejemplos

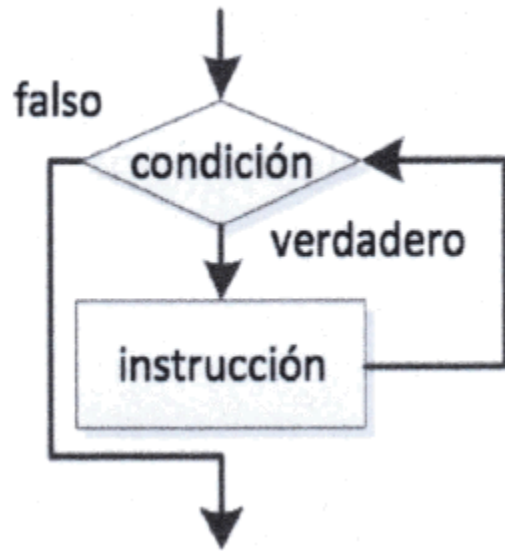




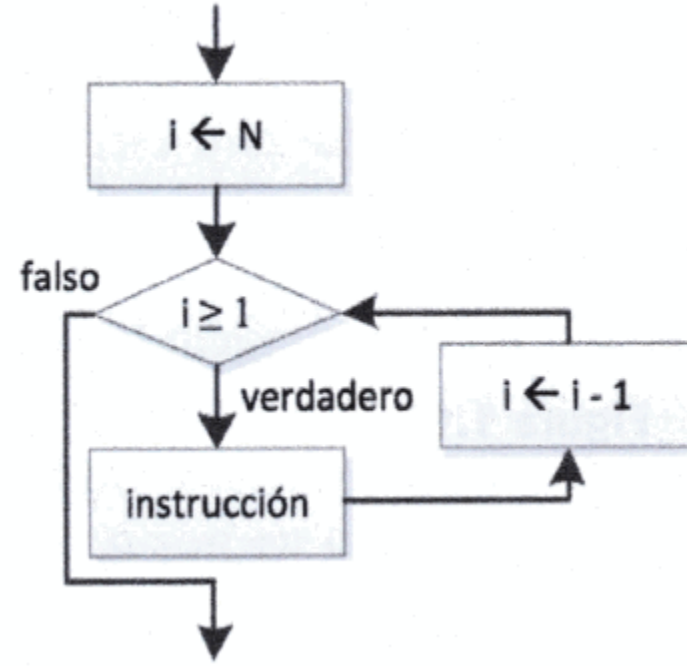
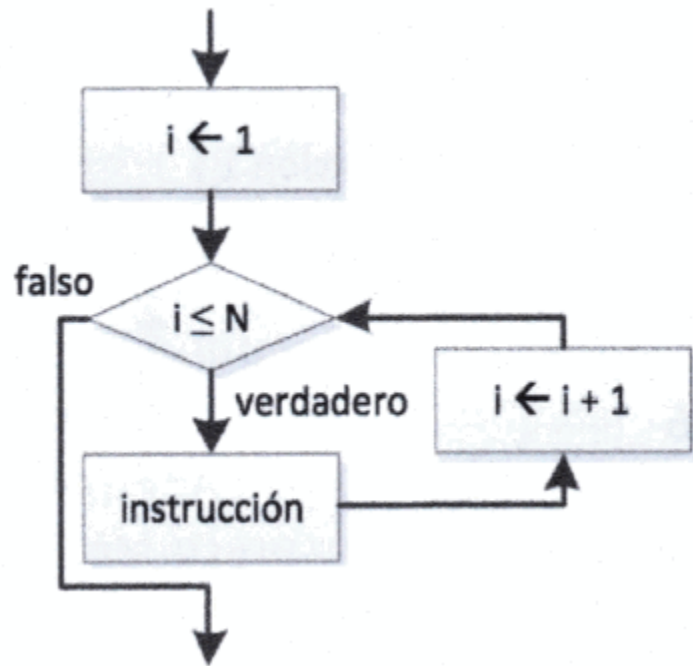
# Ejemplos



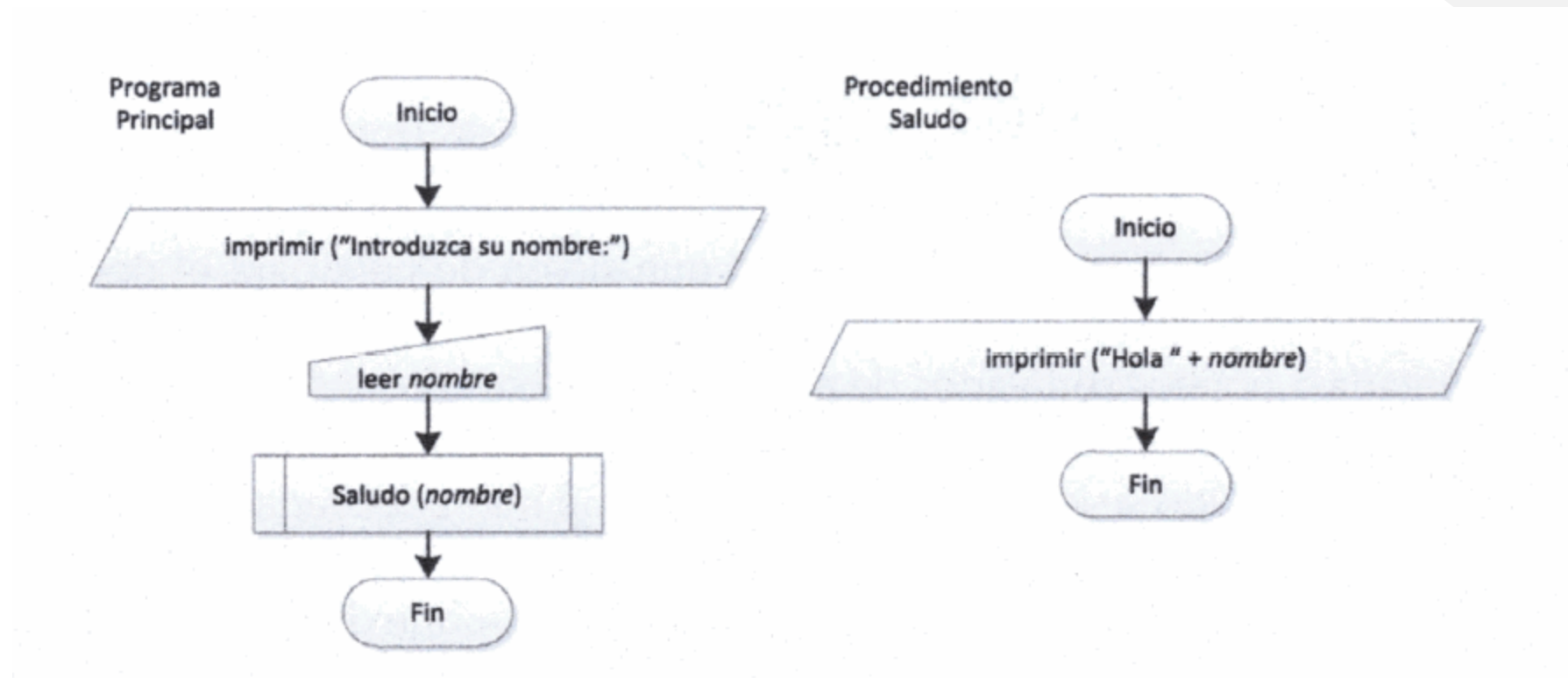
# Ejemplos



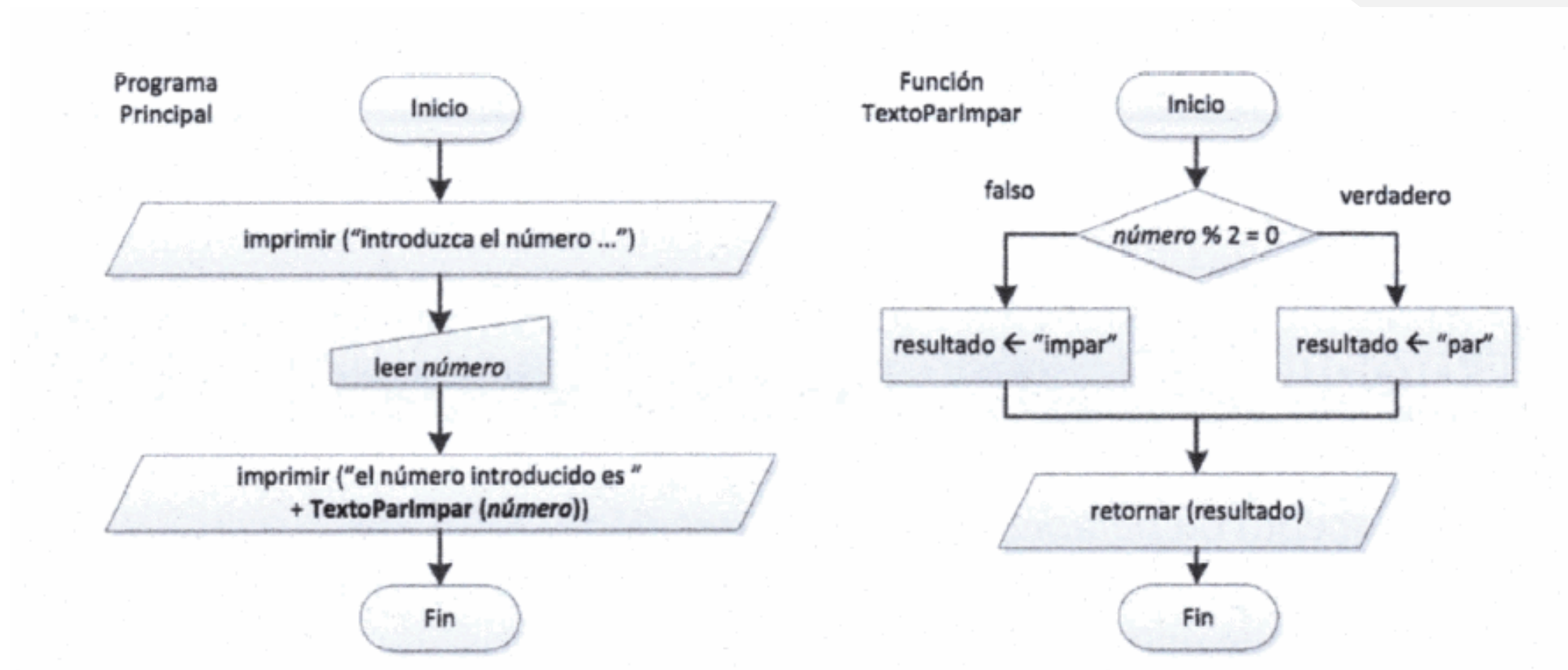
# Ejemplos



# Ejemplos



# Ejemplos



# Actividades - Ejercicios

1. Realizar la carga del lado de un cuadrado, mostrar por pantalla el perímetro del mismo (El perímetro de un cuadrado se calcula multiplicando el valor del lado por cuatro).
2. Escribir un programa en el cual se ingresen cuatro números, calcular e informar la suma de los dos primeros y el producto del tercero y el cuarto.
3. Realizar un programa que lea cuatro valores numéricos e informar su suma y promedio.

# Actividades - Ejercicios

4. Se debe desarrollar un programa que pida el ingreso del precio de un artículo y la cantidad que lleva el cliente. Mostrar lo que debe abonar el comprador.
5. Igual que el anterior, pero aplicándole al precio final un descuento del 10%. Mostrar el resultado donde aparezca el importe inicial, el descuento, el importe a pagar.
6. Programa que solicita dos números enteros por teclado, para mostrar por pantalla el cociente y el resto de la división. Para evitar excepciones, evitar que el segundo número, el divisor, sea 0.



# Actividades – Ejercicios – Solución 1 Pseudocódigo

ENTORNO

entero lado;

entero perimetroCuadrado;

INICIO

mostrar("Dime el lado del cuadrado: ");

leer lado;

$\text{perimetroCuadrado} = \text{lado} * 4;$

mostrar("El perimetro es " + perimetroCuadrado);

FIN



# Estructura de un programa

```
/**  
 * Estructura general de un programa en Java  
 */
```

```
public class Clase_Principal  
{
```

Nombre de la clase: tendrá un nombre representativo.

```
    // Definición de atributos de la clase  
    // Definición de métodos de la clase  
    // Definición de otras clases de usuario
```

```
    // Declaración del método main  
    public static void main (String[] args)  
    {
```

Cuerpo del método principal: Donde se desarrollará la ejecución.

```
        //Declaración de variables del método  
        //Instrucciones que se quieren insertar aquí...
```

```
    }  
}
```

# Estructura de un programa

- **public class Clase\_Principal:** Todos los programas han de incluir una clase como ésta. Es una clase general en la que se incluyen todos los demás elementos del programa. Entre otras cosas, contiene el método o función `main()` que representa al programa principal, desde el que se llevará a cabo la ejecución del programa. Este método es el que ejecutará en primer lugar cuando una aplicación se lance a ejecución. Esta clase puede contener a su vez otras clases del usuario, pero sólo una puede ser `public`. El nombre del fichero `.Java` que contiene el código fuente de nuestro programa, coincidirá con el nombre de la clase que estamos describiendo en estas líneas.
- **public static void main (String[ ] args):** Es el método que representa al programa principal, en él se podrán incluir las instrucciones que estimemos oportunas para la ejecución del programa. Desde él se podrá hacer uso del resto de clases creadas. Todos los programas Java tienen un método `main`.

# Estructura de un programa

- **Comentarios:** Los comentarios se suelen incluir en el código fuente para realizar aclaraciones, anotaciones o cualquier otra indicación que el programador estime oportuna. Estos comentarios pueden introducirse de dos formas, con `//` y con `/* */`. Con la primera forma estaríamos estableciendo una única línea completa de comentario y, con la segunda, con `/*` comenzaríamos el comentario y éste no terminaría hasta que no insertáramos `*/`.
- **Bloques de código:** son conjuntos de instrucciones que se marcan mediante la apertura y cierre de llaves `{` `}`. El código así marcado es considerado interno al bloque.
- **Punto y coma:** aunque en el ejemplo no hemos incluido ninguna línea de código que termine con punto y coma, hay que hacer hincapié en que cada línea de código ha de terminar con punto y coma `;`. En caso de no hacerlo, tendremos errores sintácticos.

# Utilización de las librerías básicas con Java

En Java y en varios lenguajes de programación más, existe el concepto de librerías. Una librería en Java se puede entender como un conjunto de clases, que poseen una serie de métodos y atributos. Lo realmente interesante de estas librerías para Java es que facilitan muchas operaciones. De una forma más completa, las librerías en Java nos permiten reutilizar código, es decir que podemos hacer uso de los métodos, clases y atributos que componen la librería evitando así tener que implementar nosotros mismos esas funcionalidades.

# Principales paquetes de clases

Conforme nuestros programas se van haciendo más grandes, el número de clases va creciendo. Meter todas las clases en único directorio no ayuda a que estén bien organizadas, lo mejor es hacer grupos de clases, de forma que todas las clases que estén relacionadas o traten sobre un mismo tema estén en el mismo grupo.

Un **paquete** de clases es una agrupación de clases que consideramos que están relacionadas entre sí o tratan de un tema común.

**Las clases de un mismo paquete tienen un acceso privilegiado a los atributos y métodos de otras clases de dicho paquete. Es por ello por lo que se considera que los paquetes son también, en cierto modo, unidades de encapsulación y ocultación de información.**

# Principales paquetes de clases

**Java** nos ayuda a organizar las clases en **paquetes**. En cada fichero .java que hagamos, al principio, podemos indicar a qué paquete pertenece la clase que hagamos en ese fichero.

Los paquetes se declaran utilizando la palabra clave ***package*** seguida del nombre del paquete. Para establecer el paquete al que pertenece una clase hay que poner una sentencia de declaración como la siguiente al principio de la clase:

```
package Nombre_de_Paquete;
```

# Principales paquetes de clases

## Ejemplo

The screenshot illustrates a Java project structure and a corresponding code file. On the left, the 'Projects' panel shows a project named 'Bienvenida' with a source package 'ejemplos' containing the file 'Bienvenida.java'. A red box highlights 'Bienvenida.java' in the file list, and a blue box points to it with the text 'Proyecto con un paquete: ejemplos'. The main editor window displays the code for 'Bienvenida.java'. A blue box highlights the package declaration 'package ejemplos;' with the text 'Paquete "ejemplos"'. A red oval highlights the class declaration 'public class Bienvenida {' with the text 'Clase incluida en el nuevo paquete'. The code includes a comment 'Ejemplo de uso de paquetes' and an '@author FMA' annotation. The 'main' method is shown with the line 'System.out.println ("Bienvenido a Java");' highlighted in blue.

```
1  /*
2   * Ejemplo de uso de paquetes
3   */
4
5  package ejemplos;
6
7  /**
8   *
9   * @author FMA
10  */
11  public class Bienvenida {
12
13
14      public static void main(String[] args) {
15
16          System.out.println ("Bienvenido a Java");
17
18      }
19
20  }
21
```

# Sentencia import

Cuando queremos utilizar una clase que está en un paquete distinto a la clase que estamos utilizando, se suele utilizar la sentencia import. Por ejemplo, si queremos utilizar la clase Scanner que está en el paquete java.util de la Biblioteca de Clases de Java, tendremos que utilizar esta sentencia:

```
import java.util.Scanner;
```

Se pueden importar todas las clases de un paquete, así:

```
import java.awt.*;
```



# Sentencia import

Esta sentencia debe aparecer al principio de la clase, justo después de la sentencia package, si ésta existiese.

También podemos utilizar la clase sin sentencia import, en cuyo caso cada vez que queramos usarla debemos indicar su ruta completa:

```
java.util.Scanner teclado = new java.util.Scanner (System.in);
```

# Librerías Java

Cuando descargamos el entorno de compilación y ejecución de Java, obtenemos la API de Java. Como ya sabemos, se trata de un conjunto de bibliotecas que nos proporciona paquetes de clases útiles para nuestros programas.

Utilizar las clases y métodos de la Biblioteca de Java nos va ayudar a reducir el tiempo de desarrollo considerablemente, por lo que es importante que aprendamos a consultarla y conozcamos las clases más utilizadas.

# Librerías Java

- **java.io.** Contiene las clases que gestionan la entrada y salida, ya sea para manipular ficheros, leer o escribir en pantalla, en memoria, etc. Este paquete contiene por ejemplo la clase `BufferedReader` que se utiliza para la entrada por teclado.
- **java.lang.** Contiene las clases básicas del lenguaje. Este paquete no es necesario importarlo, ya que es importado automáticamente por el entorno de ejecución. En este paquete se encuentra la clase `Object`, que sirve como raíz para la jerarquía de clases de Java, o la clase `System` que ya hemos utilizado en algunos ejemplos y que representa al sistema en el que se está ejecutando la aplicación. También podemos encontrar en este paquete las clases que "envuelven" los tipos primitivos de datos. Lo que proporciona una serie de métodos para cada tipo de dato de utilidad, como por ejemplo las conversiones de datos.
- **java.util.** Biblioteca de clases de utilidad general para el programador. Este paquete contiene por ejemplo la clase `Scanner` utilizada para la entrada por teclado de diferentes tipos de datos, la clase `Date`, para el tratamiento de fechas, etc.

# Librerías Java

- **java.math.** Contiene herramientas para manipulaciones matemáticas.
- **java.awt.** Incluye las clases relacionadas con la construcción de interfaces de usuario, es decir, las que nos permiten construir ventanas, cajas de texto, botones, etc. Algunas de las clases que podemos encontrar en este paquete son Button, TextField, Frame, Label, etc.
- **java.swing.** Contiene otro conjunto de clases para la construcción de interfaces avanzadas de usuario. Los componentes que se engloban dentro de este paquete se denominan componentes Swing, y suponen una alternativa mucho más potente que AWT para construir interfaces de usuario.
- **java.net.** Conjunto de clases para la programación en la red local e Internet.
- **java.sql.** Contiene las clases necesarias para programar en Java el acceso a las bases de datos.
- **java.security.** Biblioteca de clases para implementar mecanismos de seguridad.

# Clase Scanner

Provee métodos para leer valores de entrada de varios tipos y forma parte del paquete `java.util`. Los valores de entrada pueden venir de varias fuentes, ya sea desde teclado o desde fichero. Para utilizar esta clase deberemos importar la clase **`java.util.Scanner`** en nuestro código:

```
import java.util.Scanner;
```

Posteriormente tenemos que crear primero un objeto de ella para poder invocar sus métodos. La siguiente declaración crea un objeto llamado *teclado* de la clase `Scanner` que lee valores de entrada del teclado.

```
Scanner teclado = new Scanner(System.in);
```

# Clase Scanner

El propósito de pasar a `System.in` como argumento es conectar o establecer una relación entre el objeto tipo `Scanner`, con nombre teclado en la declaración anterior, y el objeto `System.in`, que representa el sistema estándar de entrada de información en Java. Si no se indica lo contrario, el teclado es, por omisión, el sistema estándar de entrada de información en Java.

Luego que se tenga un objeto de la clase `Scanner` asociado al sistema estándar de entrada `System.in`, llamamos, por ejemplo, su método **`nextInt()`** para entrar un valor del tipo **`int`**. Para entrar otros valores de otros tipos de datos primitivos, se usan los métodos correspondientes como **`nextByte()`** o **`nextDouble()`**.

# Clase Scanner

Método	Ejemplo
<b>nextByte()</b>	<code>byte b = teclado.nextByte();</code>
<b>nextDouble()</b>	<code>double d = teclado.nextDouble();</code>
<b>nextFloat()</b>	<code>float f = teclado.nextFloat();</code>
<b>nextInt()</b>	<code>int i = teclado.nextInt();</code>
<b>nextLong()</b>	<code>long l = teclado.nextLong();</code>
<b>nextShort()</b>	<code>short s = teclado.nextShort();</code>
<b>next()</b>	<code>String p = teclado.next();</code>
<b>nextLine()</b>	<code>String o = teclado.nextLine();</code>

# Clase Scanner

## Ejemplo

```
import java.util.Scanner;

public class EntradaDatos {

    public static void main(String[] args){

        //Se crea el lector

        Scanner sc = new Scanner(System.in);

        //Se pide un dato al usuario, en este caso su nombre

        System.out.print("Por favor introduzca su nombre: ");

        //Se lee el nombre con nextLine() que retorna un String con el dato

        String nombre = sc.nextLine();

        //Se pide otro dato al usuario, en este caso su edad

        System.out.print("Bienvenido " + nombre + ". Por favor, introduzca su edad: ");

        //Se guarda la edad directamente con nextInt()

        int edad = sc.nextInt();

        System.out.println("Te llamas " + nombre + " y tienes " + edad + " años.");

        // Cerramos el lector

        sc.close();

    }
}
```