CLASE 3. Elementos del Lenguaje de Programación Java

Introducción a los Lenguajes de Programación

Los lenguajes de programación son fundamentales en el desarrollo de software, pues permiten traducir las instrucciones humanas a un formato que las máquinas puedan ejecutar. Desde los inicios de la informática, los lenguajes de programación han facilitado la creación de soluciones tecnológicas en diversas áreas, desde aplicaciones sencillas hasta sistemas complejos.

Importancia de los Lenguajes de Programación

El uso de un lenguaje de programación es crucial en la ingeniería de software, ya que define las herramientas, el estilo y las capacidades para desarrollar aplicaciones. Algunos de los motivos por los cuales se elige un lenguaje de programación sobre otro incluyen:

- **Eficiencia**: Algunos lenguajes permiten escribir código más eficiente, optimizado para la memoria o el rendimiento del hardware.
- **Comunidad y soporte**: La existencia de una comunidad activa es vital para la resolución de problemas y el crecimiento del lenguaje.
- **Compatibilidad**: Los lenguajes de programación ofrecen distintos niveles de compatibilidad con sistemas operativos y plataformas tecnológicas.

Primeros Lenguajes de Programación

El desarrollo de los lenguajes de programación comenzó con la creación de los primeros sistemas informáticos. Los primeros lenguajes, como **Fortran** y **COBOL**, establecieron las bases de la programación estructurada. Estos lenguajes se utilizaron para resolver problemas científicos y empresariales y, aunque datan de las décadas de 1950 y 1960, aún tienen aplicaciones en sectores específicos.

Historia de los Lenguajes de Programación

Antes del año 2000: Los lenguajes creados antes del año 2000, como **C**, **C++** y **Pascal**, revolucionaron la programación al introducir paradigmas como la programación estructurada y orientada a objetos.
Después del año 2000: A partir del 2000, surgieron lenguajes más modernos como **Python**, **C#**, y **Ruby**, que pusieron énfasis en la simplicidad, la productividad y la facilidad de aprendizaje, con un enfoque más fuerte en el desarrollo ágil y la integración con nuevas tecnologías como la web y los dispositivos móviles.

Java: Un Lenguaje Universal

Historia de Java: Java fue desarrollado por **Sun Microsystems** en 1995, inicialmente diseñado para ser un lenguaje de programación sencillo, portable y seguro. Su lema “escribe una vez, ejecuta en

cualquier lugar" refleja su capacidad para ser ejecutado en múltiples plataformas sin necesidad de modificaciones al código.

Utilidades de Java:
Java es un lenguaje versátil y altamente utilizado en varias áreas de la informática:

1. **Desarrollo de aplicaciones móviles**: Es el principal lenguaje utilizado para desarrollar aplicaciones en **Android**.
2. **Inteligencia Artificial**: Se utiliza en sistemas de IA debido a su escalabilidad y capacidad para manejar grandes volúmenes de datos.
3. **Big Data**: Java es fundamental en proyectos de **Big Data** debido a su compatibilidad con marcos como **Hadoop**.
4. **Aplicaciones de Escritorio**: Muchos programas de uso cotidiano, como **NetBeans** y **Eclipse**, están desarrollados en Java.
5. **Desarrollo Web**: Con **Spring** y **JavaServer Pages (JSP)**, Java es una opción popular para aplicaciones web de gran escala.
6. **Aplicaciones IoT (Internet de las Cosas)**: Se usa en la programación de dispositivos inteligentes y sensores conectados.
7. **Juegos**: Java también es utilizado en el desarrollo de videojuegos, especialmente para la web y dispositivos móviles.

Diferencias clave entre Java y JavaScript:
Aunque sus nombres son similares, Java y JavaScript son lenguajes fundamentalmente diferentes:

- **Java** es un lenguaje orientado a objetos utilizado principalmente en aplicaciones de backend y de escritorio.
- **JavaScript** es un lenguaje interpretado, utilizado principalmente en el frontend de aplicaciones web para agregar interactividad a

las páginas.

Características Técnicas de Java

Edición, Compilación y Ejecución de un Programa Java:
Un programa Java sigue un ciclo bien definido desde su creación hasta su ejecución:

1. **Edición**: El código fuente se escribe en archivos con extensión `.java`.
2. **Compilación**: El código fuente se compila en **bytecode** utilizando el compilador `javac`. El bytecode es un formato intermedio que puede ser ejecutado en cualquier máquina con una **Java Virtual Machine (JVM)**.
3. **Ejecución**: El bytecode es interpretado o compilado en tiempo de ejecución por la JVM, permitiendo que el programa funcione en diferentes sistemas operativos sin cambios.

JVM (Java Virtual Machine):
La JVM es el corazón de la portabilidad de Java. Actúa como un intermediario entre el bytecode generado por el compilador y el sistema operativo, garantizando que los programas Java puedan ejecutarse en cualquier plataforma.

Manipulación de Cadenas de Texto en Java

Concatenación de Cadenas:
La concatenación de cadenas es el proceso de unir dos o más secuencias de caracteres. En Java, esto se puede hacer utilizando el operador `+` o el método `concat()` de la clase `String`.

Métodos de la clase `String`:
La clase `String` en Java proporciona una serie de métodos útiles para manipular cadenas de texto, como:

- `length()` para obtener la longitud de la cadena.
- `substring()` para extraer subcadenas.
- `toLowerCase()` y `toUpperCase()` para cambiar el caso de los caracteres.

Normas de Codificación en Java

Comentarios:
El uso de comentarios es esencial para hacer el código más legible. Existen tres tipos de comentarios en Java:

- **Comentarios de una sola línea**: `// Este es un comentario`
- **Comentarios de varias líneas**: `/* Este es un comentario multilinea */`
- **Comentarios de documentación**: `/** Este es un comentario de documentación */`

Nomenclatura:
La nomenclatura en Java sigue convenciones específicas para mantener la legibilidad y consistencia:

- **Nombres de clases**: Comienzan con mayúscula y usan notación **CamelCase**. Ejemplo: `NombreDeClase`.
- **Nombres de variables y métodos**: Comienzan con minúscula y usan notación camelCase. Ejemplo: `nombreDeVariable`.

Tipos de Datos en Java

¿Qué es la información?
La información es el conjunto de datos organizados que tienen un significado y se pueden procesar para obtener conocimiento útil.

¿Qué son los datos?
Los datos son representaciones simbólicas de hechos y cifras que, por sí solos, no tienen un significado inherente, pero que pueden ser interpretados cuando se organizan de manera estructurada.

¿Qué es una constante?
Una constante es un valor que no cambia durante la ejecución del programa. En Java, las constantes se definen utilizando la palabra clave `final`.

Variables en Java

Declaración de Variables:
Las variables en Java son espacios en memoria que almacenan datos. Para declarar una variable, se especifica el tipo de dato seguido del nombre de la variable. Ejemplo:

```
int edad;
```

Nombres de Variables Prohibidos:
No se pueden utilizar palabras reservadas del lenguaje como nombres de variables, tales como `class`, `static`, `void`, entre otras.

Asignación de Variables:
Una vez declarada una variable, se le puede asignar un valor

utilizando el operador `=`. Ejemplo:

```
edad = 25;
```

Tipos de Datos:
Java ofrece varios tipos de datos para almacenar diferentes valores. Los principales tipos de datos incluyen:

- **int**: Para enteros.
- **double**: Para números decimales.
- **boolean**: Para valores booleanos (`true` o `false`).
- **char**: Para caracteres individuales.