

Lenguajes de programación.  
Tipos. Características

## **TEMA 24 (26 SAI)**

---

**ABACUS NT**

Oposiciones 2021

## **Índice**

---

- 1. Introducción.**
- 2. Lenguajes de programación**
  - 2.1. Definición**
  - 2.2. Generaciones**
- 3. Tipos**
  - 3.1. Lenguajes compilados vs Lenguajes interpretados**
    - 3.1.1. Lenguajes interpretados
    - 3.1.2. Lenguajes Compilados
  - 3.2. En función del nivel de abstracción**
  - 3.3. En función paradigma de programación**
    - 3.3.1. Lenguajes Imperativos.
    - 3.3.2. Lenguajes Declarativos
    - 3.3.3. Orientados a objetos.
    - 3.3.4. Lenguajes Concurrentes.
    - 3.3.5. Lenguajes conducidos por evento
    - 3.3.6. Lenguajes Orientados a aspectos
    - 3.3.7. Programación multiparadigma
  - 3.4. Clasificación en función del dominio de aplicación**
- 4. Características de un lenguaje de programación**
  - 4.1. Características del diseño**
  - 4.2. Aspectos de la implementación**
- 5. Los lenguajes de programación actuales y su tendencia**
  - 5.1. Lenguajes más utilizados en el sector productivo**
- 6. Lenguajes para la Web**
- 7. Conclusión**
  - 7.1. Relación del tema con el sistema educativo actual**
- 8. Bibliografía**

## 1. Introducción.

**Programar** viene a ser el proceso de crear un software fiable mediante la escritura, prueba, depuración, compilación o interpretación, y mantenimiento del código fuente de dicho programa informático. Básicamente, este proceso se define aplicando lógicamente los siguientes pasos:

- El desarrollo lógico del programa para resolver un problema en particular
- Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa)
- Compilación o interpretación del programa hasta convertirlo en lenguaje de máquina
- Prueba y depuración del programa
- Desarrollo de la documentación.

Los lenguajes de programación están formados por un conjunto de símbolos (llamado alfabeto), reglas gramaticales (léxico/morfológicas y sintácticas) y semánticas, que en conjunto definen las estructuras válidas del lenguaje y su significado. **Los lenguajes informáticos engloban a los lenguajes de programación y a otros más, como por ejemplo HTML** (lenguaje para el marcado de páginas web que no es propiamente un lenguaje de programación, sino un conjunto de instrucciones que permiten estructurar el contenido de los documentos).

## 2. Lenguajes de programación

### 2.1. Definición de lenguaje de programación

Un lenguaje de programación es un **lenguaje formal** conformado por un conjunto predefinido de **palabras** y **símbolos** que se utilizan siguiendo unas reglas prefijadas (**sintaxis**), para representar y definir los **algoritmos** y **datos** que constituyen un **programa**.

Un programa está formado a su vez por un conjunto ordenado de instrucciones (símbolos que representan una orden de operación) que indican al ordenador las tareas que se desea que realice.

El lenguaje de programación permite especificar de manera precisa **sobre qué datos** debe operar un software específico, cómo deben ser almacenados o transmitidos dichos datos, y **qué acciones** debe tomar el software bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar **relativamente próximo al lenguaje humano** o natural.

### 2.2. Generaciones

Los **factores** más importantes que han influido en la evolución de los lenguajes, desde 1950, han sido:

- Evolución del **hardware** y los **sistemas operativos**
- Aumento del número de **aplicaciones**

- Nuevos **métodos** de programación
- Nuevas **herramientas** para la implementación
- Investigación y estudios **teóricos**
- **Estandarización**

Otra manera de clasificar los lenguajes es separándolos en generaciones, las cuales, según indica el autor J.Glenn Brookshear (2012), han evolucionado de la siguiente forma:

### Lenguajes máquina (primera generación)

Consiste en un conjunto limitado de instrucciones y una serie de datos codificados en binario que el procesador es capaz de ejecutar directamente

Un programa escrito en cualquier lenguaje de programación debe transformarse en lenguaje máquina para poder ser ejecutado, lo que hace necesaria la utilización de traductores

Se caracteriza por lo siguiente:

- Las instrucciones están definidas en los circuitos del procesador y realizan operaciones simples. Los datos y el destino de las instrucciones de salto de ejecución se refieren por medio de su posición en la memoria, y por tanto es necesario conocer su dirección
- El lenguaje máquina es particular de cada procesador y por tanto no es portable. Es muy difícil y costoso de programar y mantener, pero el código es totalmente eficiente

### Lenguajes ensambladores (segunda generación)

Son lenguajes simbólicos que asignan a cada instrucción del lenguaje máquina un código nemotécnico formado por caracteres alfanuméricos

Se caracterizan por lo siguiente:

- Se facilita la escritura y depuración de los programas, pero no los acorta. Se emplean etiquetas para referenciar cualquier parte del código y nombres para referenciar a los datos, no siendo por tanto necesario conocer su dirección. También permite el uso de comentarios
- Al tratarse de transformaciones del lenguaje máquina, los lenguajes ensambladores son también particulares de cada procesador y por tanto no son portables
- Es menos costoso que el lenguaje máquina, pero igualmente exige un total conocimiento del hardware. Está indicado sobre todo para programar tareas relacionadas con el control de los periféricos y la gestión de la memoria

Debido a que la programación de cualquier tarea debe ser descompuesta en multitud de pasos elementales, para aumentar la eficiencia en la programación se recurre a crear instrucciones denominadas macroinstrucciones que agrupen a varias instrucciones de código máquina. El lenguaje así construido se denomina macroensamblador

## Procedimentales o Procedurales (tercera generación)

Son lenguajes más cercanos al lenguaje humano en los cuales la solución al problema se describe paso a paso mediante un conjunto de instrucciones más flexibles y potentes

Se caracterizan por lo siguiente:

- Son independientes del hardware, y por tanto son portables. En la práctica, esta característica está limitada por la gran diversidad de versiones que existen de cada lenguaje
- Una instrucción en estos lenguajes da lugar a varias instrucciones en lenguaje máquina
- Son más lentos y menos eficientes que los lenguajes ensambladores ya que deben ser traducidos a código máquina, pero son más sencillos de codificar y mantener
- A diferencia de los lenguajes ensambladores, soportan la programación estructurada

Entre los lenguajes procedimentales se pueden destacar actualmente Modula-2, C, C++ y Java

## No procedimentales (cuarta generación)

Son lenguajes en los que el programador no especifica el procedimiento a seguir, puesto que el propio lenguaje es capaz de indicar al ordenador cómo debe ejecutar el programa

Se caracterizan por lo siguiente:

- El programador debe definir una serie de parámetros que, mediante la utilización de un conjunto de herramientas del lenguaje, permite generar un programa de aplicación

- Son más fáciles de utilizar que los de tercera generación, ya que suelen incluir interfaces gráficos y capacidades de gestión avanzadas, pero consumen muchos más recursos
- No requieren una capacitación especial por parte del programador, mejorando su productividad, y están diseñados para resolver problemas específicos

En los lenguajes de cuarta generación están incluidos:

- Lenguajes de presentación, como lenguajes de consultas y generadores de informes
- Lenguajes especializados, como hojas de cálculo y lenguajes de bases de datos
- Generadores de aplicaciones que definen, actualizan y obtienen datos de una base de datos
- Lenguajes utilizados para generar el código de una aplicación

Entre este tipo de lenguajes se pueden destacar actualmente SQL, Visual Basic y Forms

## Naturales (quinta generación)

Son lenguajes cuyos programas están constituidos por hechos, reglas, restricciones, ecuaciones, transformaciones, u otras propiedades que debe cumplir la solución al problema.

Se caracterizan por lo siguiente:

- A partir de la información introducida por el programador, el sistema debe deducir un esquema que incluya una secuencia de evaluaciones para calcular la solución
- Están orientados a aplicaciones en inteligencia artificial, es decir, sistemas basados en el conocimiento, sistemas expertos o procesamiento del lenguaje natural

Ejemplos de estos lenguajes son LISP y PROLOG, que usan técnicas de inteligencia artificial.

### Lenguajes Visuales y orientados a objetos (Sexta generación)

A la clasificación de J.Glenn Brookshear se le debe añadir una generación extra dado el gran avance que la programación visual con orientación a objetos tiene en la actualidad.

Los lenguajes de programación visual permiten a los usuarios crear programas mediante la manipulación de elementos gráficos, en lugar de especificarlos exclusivamente de manera textual.

Sus características son:

- Incorporan una dimensión visual para expresar la semántica del lenguaje.
- Son lenguajes orientados a objetos o multiparadigma.
- Expresan un programa como un conjunto de objetos que colaboran entre ellos para realizar tareas.

Ejemplos son: VisualBasic.net Borland Delphi y Scratch.

## 3. Tipos

Existe una amplia lista de lenguajes de programación, como hemos visto en el apartado anterior, los cuales pueden ser clasificados de diferentes maneras según los siguientes criterios:

### 3.1. Lenguajes compilados vs Lenguajes interpretados

Al contrario de lo que puede parecer en un principio, ningún lenguaje de programación es entendible directamente por el ordenador ni siquiera el código ensamblador. Todo debe ser **traducido a lenguaje máquina**.

Este proceso de **traducción**, va a partir de un **código fuente**, en el que hemos codificado el programa que queremos ejecutar y va a obtener un **código objeto** en lenguaje máquina comprensible para una arquitectura de ordenador determinada.

La traducción se puede llevar a cabo de dos maneras radicalmente distintas que van a determinar aspectos esenciales del lenguaje; el código fuente puede ser “compilado” a código objeto antes de su ejecución, o puede ser traducido instrucción a instrucción justo en el preciso momento de su ejecución, dando lugar a dos tipos de lenguajes distintos:

### 3.1.1. Lenguajes interpretados

Un intérprete es un programa que traduce y ejecuta un programa fuente instrucción a instrucción por lo que una instrucción no será traducida hasta que no se ejecute la anterior. No genera programa objeto lo cual obliga a traducir el programa cada vez que se deseé ejecutar. Presenta la ventaja de que la ejecución se realiza inmediatamente, lo que permite detectar rápidamente errores y corregirlos en el momento. Sin embargo, tienen el inconveniente de ralentizar innecesariamente la ejecución del programa si éste incluye bucles o llamadas a funciones.

### 3.1.2. Lenguajes Compilados

Un compilador es un programa que traduce un código fuente en bloque, generando un programa en lenguaje máquina llamado programa objeto. Este proceso requiere más tiempo en la traducción que el intérprete. El programa no se ejecuta hasta que se haya traducido entero. Por el contrario, una vez traducido, la ejecución será mucho más rápida.

En el caso de que el lenguaje sea de bajo nivel, esto es lenguaje ensamblador, el traductor utilizado también se denomina ensamblador.

## 3.2. En función del nivel de abstracción

En función del nivel de abstracción podemos diferenciar entre:

- **Lenguaje máquina:** En este caso existe una dependencia total con la arquitectura física de la máquina. Estos lenguajes siguen una notación binaria.
- **Lenguajes ensambladores:** Permiten escribir instrucciones con una notación simbólica o nemotécnica.
- **Lenguajes de alto nivel:** Aportan una mayor independencia de la máquina. Como ejemplo tenemos Haskell.

## 3.3. En función paradigma de programación

Según el autor Luis Joyanes Aguilar, en su libro Fundamentos de programación. Algoritmos, estructuras de datos y objetos (2008), “un paradigma de programación **representa** el **enfoque** que se utiliza para obtener una **solución a un problema**, el cual **afecta** al proceso completo del **desarrollo del software**”. Los lenguajes se pueden clasificar en los siguientes paradigmas:

### 3.3.1. Lenguajes Imperativos.

Se basan en un modelo de ordenador como una máquina que almacena datos y un conjunto de instrucciones ejecutadas secuencialmente que se encargan de modificar dichos datos. Están formados por sentencias agrupadas en procedimientos, cada uno de los cuales realiza una tarea concreta y se relaciona con el resto del programa mediante llamadas con paso de parámetros.

En su forma más pura sólo soportan la modificación de datos mediante la utilización de variables y sentencias de asignación, sentencias de salto condicional y de salto

incondicional. Son más eficientes porque se asemejan al modo de funcionamiento de la máquina.

Entre los lenguajes imperativos se pueden destacar Modula-2 y C.

### **3.3.2. Lenguajes Declarativos**

Un lenguaje declarativo es un tipo de lenguaje de programación basado más en las matemáticas y en la lógica que los lenguajes imperativos, más cercanos estos al razonamiento humano. Los lenguajes declarativos no dicen cómo hacer una cosa, sino, más bien, qué cosa hacer. Hay dos subtipos:

#### **Lenguajes Funcionales.**

Se trata de lenguajes de programación sin asignaciones, basados en el concepto matemático de función. Consisten en combinar funciones para conseguir funciones más complejas hasta llegar a la función que es el programa. Los valores de las funciones son también funciones.

Ejemplos de estos lenguajes son LISP y SCHEME.

#### **Lenguajes Lógicos.**

Son lenguajes en los que se especifican un conjunto de hechos y una serie de reglas que permiten la deducción de otros hechos. El sistema utiliza esa información para encontrar la solución.

La programación lógica desde la perspectiva del programador consiste en establecer correctamente todos los hechos y reglas, ya que el cálculo está implícito.

El lenguaje lógico más representativo es PROLOG, basado en la lógica de predicados.

### **3.3.3. Orientados a objetos.**

Es un tipo de programación de un nivel de abstracción mayor, que consiste en modelar la realidad como un conjunto de objetos que interactúan entre sí para resolver un problema.

Las características básicas comunes que presentan los lenguajes orientados a objetos son:

- **Encapsulamiento.** Las propiedades y el comportamiento de los objetos están definidos en las clases, formadas por estructuras de datos y algoritmos denominadas atributos y métodos. Un objeto es una instancia de una determinada clase y tiene su propio conjunto de datos. Los métodos se aplican a un objeto concreto y son propias de la clase a la que pertenece.

- **Ocultación.** Para aumentar la modularidad y como medida de protección, un objeto se considera como una caja negra en la que se puede introducir o de la que se puede extraer información sin necesidad de conocer su funcionamiento interno.

- **Envío de mensajes.** Consiste en nombrar y activar con los parámetros adecuados uno de los métodos del objeto al que se envía el mensaje. Es la manera que tienen los

objetos de relacionarse, y es equivalente a las llamadas a procedimientos de los lenguajes imperativos.

- **Herencia.** Es un mecanismo mediante el cual pueden crearse clases a partir de otras, transmitiéndose todos los atributos y todos los métodos de clases padres a clases hijas, con la ventaja de que en estas últimas pueden añadirse más atributos y métodos.
- **Polimorfismo.** Es la capacidad que tiene un objeto de una determinada clase de hacerse pasar por un objeto de una clase ancestro. Esta característica, junto a la posibilidad de las clases hijas de redefinir los métodos heredados, permiten la reutilización de código.

Entre los lenguajes orientados a objetos se pueden destacar actualmente C++ y Java.

### **3.3.4. Lenguajes Concurrentes.**

Son lenguajes cuyo elemento fundamental es el proceso, que es una instancia de un programa que está siendo ejecutado por un ordenador junto con sus datos asociados y los recursos que utiliza. Un hilo de un programa concurrente es cada uno de los flujos secuenciales de control independientes especificados en dicho programa.

Los programas concurrentes dan lugar a un proceso con varios hilos de ejecución. La concurrencia puede ser física, cuando existe más de un procesador y varios hilos de un mismo programa se ejecutan realmente de forma simultánea, o lógica, cuando existe un único procesador y hay que repartir su tiempo entre los distintos hilos. Hay dos modelos de ejecución:

- **Transformativo.** Cuando el fin consiste en transformar los datos de entrada en valores de salida apropiados, y la concurrencia se aplica para acelerar el proceso.
- **Reactivo.** Cuando se producen sucesos externos asíncronos, como en los sistemas de tiempo real, y la concurrencia se aplica para que el programa reaccione ante dichos sucesos.

Un lenguaje de programación es concurrente si posee las estructuras necesarias para definir y manejar diferentes hilos de ejecución dentro de un programa, por ejemplo, Java y ADA.

### **3.3.5. Lenguajes conducidos por evento**

Son escritos para reaccionar a cualquier evento, una vez comienza su ejecución. Como ejemplo tenemos Swift.

### **3.3.6. Lenguajes Orientados a aspectos**

En este paradigma se separan las funcionalidades del sistema, existiendo funcionalidades transversales (aspectos). Como ejemplo tenemos AspectJ (es una extensión de Java).

### 3.3.7. Programación multiparadigma

Es el uso de dos o más paradigmas dentro de un programa. El lenguaje Lisp se considera multiparadigma. Al igual que Python, que es orientado a objetos, reflexivo, imperativo y funcional. Según lo describe Bjarne Stroustrup, esos lenguajes permiten crear programas usando más de un estilo de programación.

El objetivo en el diseño de estos lenguajes es permitir a los programadores utilizar el mejor paradigma para cada trabajo, admitiendo que ninguno resuelve todos los problemas de la forma más fácil y eficiente posible. Por ejemplo, lenguajes de programación como C++, Genie, Delphi, Visual Basic, PHP o D combinan el paradigma imperativo con la orientación a objetos. Incluso existen lenguajes multiparadigma que permiten la mezcla de forma natural, como en el caso de Oz, que tiene subconjuntos (particularidad de los lenguajes lógicos), y otras características propias de lenguajes de programación funcional y de orientación a objetos. Otro ejemplo son los lenguajes como Scheme de paradigma funcional o Prolog (paradigma lógico), que cuentan con estructuras repetitivas, propias del paradigma imperativo.

## 3.4. Clasificación en función del dominio de aplicación

Podemos tener la siguiente clasificación en función de los dominios de aplicación siguientes:

- **Aplicaciones científico-técnicas:** Como ejemplo tenemos MATLAB.
- **Aplicaciones de gestión de información:** Como ejemplo tenemos Java, o SQL.
- **Aplicaciones de programación de sistemas:** Como ejemplo tenemos C, o C++.
- **Aplicaciones Web:** Como ejemplo tenemos PHP, o Javascript.
- **Aplicaciones para movilidad:** Como ejemplos tenemos Swift, o Kotlin.

## 4. Características de un lenguaje de programación

### 4.1. Características del diseño

Las características deseables en un lenguaje de programación, según Kenneth C.Louden (2004) son las siguientes:

#### Eficiencia en la ejecución

Al iniciarse los lenguajes de programación, existía un criterio de diseño primordial: eficiencia en la ejecución. Este principio puede abarcar características para la eficiencia del código:

- 1.- **Optimización:** se refiere al diseño del lenguaje que debe ser tal que un traductor nos pueda generar un código ejecutable eficiente.
- 2.- **Eficiencia de traducción:** se refiere a la verificación de errores que podrían tener al compilar/interpretar el código fuente y podría resultar en que el código objeto resulte ineficiente.

3.- **La confiabilidad:** se refiere al aseguramiento de que un programa no se comportará en forma no esperada o desastrosa durante la ejecución. Un programa que no sea confiable genera muchos costos adicionales como causar un completo desperdicio de tiempo de desarrollo y de codificación.

4.- **La capacidad de la implementación:** es la eficiencia con la cual se puede escribir un traductor. El éxito de un lenguaje se puede obstaculizar simplemente porque es demasiado difícil escribir un traductor (compilador o intérprete).

5.-**Eficiencia de la programación:** esto está claramente relacionada con la potencia y la generalidad de los mecanismos de abstracción del lenguaje. Lo conciso de la sintaxis y evitar detalles innecesarios, como son las declaraciones de variables, a menudo se consideran también factores importantes en este tipo de eficiencia.

6.- **La capacidad de darle mantenimiento:** se refiere a la facilidad con la cual se pueden localizar los errores y corregirse, así como agregarse nuevas características.

## Regularidad

Implica las restricciones no usuales en el uso de constructores particulares también menos restricciones raras entre dichos constructores. Se subdivide en 3 conceptos más precisos:

1.- **generalidad:** se refiere al eliminar casos especiales en la disponibilidad en el uso de constructores y combinar constructores íntimamente relacionados en uno solo más general.

2.- La **ortogonalidad:** significa que los constructores de los lenguajes se pueden combinar en cualquier forma significativa y que la interacción de los constructores, o el contexto del uso, no debe generar restricciones o comportamientos inesperados.

3.- La **uniformidad:** significa que cosas similares deben verse de manera similar y tener significados similares y, a la inversa, las cosas diferentes deben verse diferentes.

## Simplicidad

Un lenguaje de programación demasiado simple puede, de hecho, hacer que la tarea de utilizarlo resulte más compleja por la carencia de algunos constructores fundamentales.

## Expresividad

La expresividad es la facilidad con la cual un lenguaje puede expresar procesos y estructuras complejas. Algunas veces la expresividad se considera poco concisa, lo que puede, por tanto, comprometer la legibilidad.

## Extensibilidad

Permite que el usuario pueda agregar características a un lenguaje.

## Capacidad de restricción

En el diseño del lenguaje conceptos estándares como son los programas, funciones y variables deben quedar claramente reconocibles.

## Precisión

Es la existencia de una definición precisa para un lenguaje, de tal manera que el comportamiento de los programas pueda ser predecible.

## Independencia de la maquina

El diseño de un lenguaje debe intentar aislar e identificar todas aquellas dependencias de la máquina que no se pueden evitar, de manera que el programador sepa exactamente dónde pueden presentarse dificultades.

## Seguridad

Promueve un diseño de lenguaje que permite que los errores sean descubiertos e informados. La seguridad está íntimamente relacionada con la confiabilidad y con la precisión.

## 4.2. Aspectos de la implementación

En cuanto a las características relacionadas con la implementación, tenemos:

### **Curva de aprendizaje:**

La curva de aprendizaje se define como el grado de éxito obtenido durante el aprendizaje en el transcurso del tiempo. Una curva pronunciada implica un aprendizaje rápido.

**Compilado o interpretado:** Según el contexto de implementación será mejor un sistema de traducción u otro.

**Portabilidad:** facilidad de recompilar el código para distintos sistemas operativos y/o ordenadores.

**Soporte externo y documentación:** Base de conocimiento, manuales y ayuda por parte del desarrollador o la comunidad.

**Coste de mantenimiento:** Teniendo en cuenta que dos tercios del tiempo del ciclo de vida del software se gasta en mantenimiento, este es un aspecto relevante.

## 5. Los lenguajes de programación actuales y su tendencia

## 5.1. Lenguajes más utilizados en el sector productivo

Según la publicación “**2019 The top Programming Languages**” del IEEE Spectrum (el espectro del Instituto de ingeniería eléctrica y electrónica), de los diez lenguajes con más importancia en el sector productivo, 9 son multiparadigma y/o orientados a objetos, lo que muestra la gran importancia de la P.O.O actualmente. Estos lenguajes son:

1. **Python:** Desarrollado por Guido Van Rossum que liberó su código en 1991 bajo licencia Python License originalmente y GNU GPL en la actualidad, es un lenguaje de programación interpretado, multiplataforma, que soporta P.O.O, programación imperativa y funcional. Hace hincapié en la legibilidad del código, utilizando un tipado dinámico y conteo para el administrador de memoria. Debe su nombre a la aficción de su desarrollador por los Monthly Python.
2. **Java:** Fue desarrollado en 1985 y está enfocado hacia la red, debido al auge de Internet en los años 90. Es independiente del ordenador donde se ejecute y para conseguir esta independencia utiliza una máquina virtual java (JVM) que nos permite utilizar código compilado en otra computadora distinta. Entre sus características destacamos:
  - Sintaxis similar a C++
  - Dispone de un recolector de basura
  - Elimina el uso de punteros, utilizando referencias
  - Soporta programación multihilo
  - Dispone de mecanismos de detección de errores
  - Tiene amplias librerías de clases
3. **C:** Desarrollado en 1972 a partir de otro lenguaje llamado B, con la idea de conseguir un lenguaje de alto nivel, pero con posibilidad de acceso a bajo nivel. Esta propiedad hace que este lenguaje sea muy adecuado para la programación de sistemas.

Es adecuado para programadores expertos porque hace programas muy eficientes, por el contrario, su depuración puede ser bastante compleja. Tiene las siguientes características:

- Es débilmente tipado
  - Impone pocas restricciones al programador
  - Dispone de operadores a nivel de bit
  - Soporta el uso de punteros
  - Utiliza librerías
4. **C++:** Fue inventado en los laboratorios de AT&T en 1980. Inicialmente fue una gran mejora de C. Actualmente tiene un estándar, denominado ISO C++.

C++ soporta programación orientada a objetos incluyendo características como:

- Implementación de clases
  - Herencia múltiple
  - Acceso a los atributos y métodos
5. **R:** Es un entorno y lenguaje de programación con un enfoque al análisis estadístico.

Se trata de uno de los lenguajes de programación más utilizados en investigación científica, junto con Matlab, siendo además muy popular en los campos de aprendizaje automático (machine learning), minería de datos, investigación biomédica, bioinformática y matemáticas financieras. R es orientado a objetos, se puede integrar con bases de datos, tiene numerosas funciones de cálculo y funciones gráficas compatibles con LaTeX.

## 6. JavaScript:

Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX.

## 7. C#:

Pronunciado C sharp, es un lenguaje de programación multiparadigma desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, de los lenguajes de programación diseñados para la infraestructura de lenguaje común (CLI).

Su sintaxis básica deriva de C/C++, utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes.

## 8. Matlab:

Matrix Laboratory es un sistema de cómputo numérico que ofrece un IDE y lenguaje de programación llamado M que es interpretado y orientado a objetos. Este lenguaje permite operaciones de vectores y matrices, funciones, cálculo lambda y representación de gráficos 2D y 3D.

## 9. Swift

Swift es un lenguaje de programación **multiparadigma** creado por **Apple** enfocado en el desarrollo de **aplicaciones para iOS y macOS**. Fue presentado en la WWDC 2014 y puede usar cualquier biblioteca programada en Objective-C y llamar a funciones de C.

## 10. Go

Desarrollado por Google en 2009, la concurrencia, la sencillez, tipado estático, y el no utilizar excepciones son las principales características de Go. Es el lenguaje de programación de sistemas multiproceso más eficaz existente en la actualidad, inspirado en la sintaxis de C. Soporta orientación a objetos parcialmente, ya que no implementa mecanismos de herencia y las clases son declaradas como componentes separados de Interfaces y estructuras.

## 6. Lenguajes para la Web

Son lenguajes interpretados como JavaScript o VBScript, o lenguajes especiales para uso en servidores como ASP, JSP o PHP. Estos lenguajes se mezclan con el código HTML o lo generan, para crear páginas web en el cliente y manejar información desde el servidor.

Características:

- Algunos de ellos no se consideran lenguajes de programación sino lenguajes informáticos o lenguajes de marcas, este es el caso de HTML, XML o CSS.
- Utilizan lenguajes preexistentes para gestionar la programación del servidor mediante CGI (Common Gateway Interface) o se desarrollan lenguajes específicos.
- Se utilizan lenguajes de script para la programación del lado del cliente (VBScript, JavaScript)
- La mayoría de los lenguajes utilizados son interpretados (este es el caso de todos los lenguajes de script)
- Hacen uso de técnicas como Ajax, que permiten la interacción del cliente con el servidor.

Se pueden utilizar una amplia variedad de lenguajes de programación, incluyendo C y C++ para escribir aplicaciones Web (CGI). Sin embargo, algunas herramientas de programación son, particularmente útiles:

- CGI Interfaz de entrada común (en inglés Common Gateway Interface, abreviado CGI) Es un mecanismo de comunicación entre el servidor web y una aplicación externa para crear contenido web dinámico. En una aplicación CGI, el servidor web pasa las solicitudes del cliente a un programa externo.
- HTML, técnicamente es un lenguaje de descripción de páginas más que un lenguaje de programación. Es el elemento clave para la programación en la Web
- CSS: Cascade Style Sheets (Hojas de estilo en cascada) Definen el estilo de los elementos que componen la página web.
- JavaScript, es un lenguaje interpretado de guionado (scripting) que facilita a los diseñadores de páginas Web añadir guiones a páginas Web y modos para enlazar esas páginas
- VBScript, la respuesta de Microsoft a JavaScript basada en VisualBasic (en desuso)
- Perl, lenguaje interpretado de guionado (scripting) idóneo para escritura de texto (en desuso)
- XML, lenguaje de etiquetas que resuelve todas las limitaciones de HTML dando lugar a la versión 5 del famoso lenguaje de marcas.

- AJAX, técnica de programación que permite hacer llamadas dinámicas entre el cliente y el servidor poniendo en comunicación JavaScript con el servidor (por ejemplo, en PHP)

## 7. Conclusión

Los lenguajes de programación han recorrido un largo camino desde los primeros lenguajes ensambladores hasta la actualidad. Hoy en día los principales desarrollos apuntan hacia la Inteligencia artificial como nuevo paradigma de programación. El desarrollo del lenguaje natural, la conducción automática, el reconocimiento facial y otros avances significativos de la informática vienen de la mano del desarrollo de nuevos chips especializados y de nuevos motores de inteligencia.

Este es el caso del motor Tensor Flow de Google o el GPT-3 de Elon Musk (OpenAI, Tesla).

Tensor Flow es una biblioteca de código abierto para aprendizaje automático a través de un rango de tareas, y desarrollado por Google para satisfacer sus necesidades de sistemas capaces de construir y entrenar redes neuronales para detectar y descifrar patrones y correlaciones, análogos al aprendizaje y razonamiento usados por los humanos.

GPT-3 de OpenAI es, por otro lado, un modelo de generación de lenguaje, esto significa que su objetivo es generar conversaciones, artículos, discursos, etc, en función de datos previos dados en casos de ejemplo. Puedes por ejemplo escribir dos o tres frases de un artículo y GPT-3 se encargará de escribir el resto del artículo. También puedes generar conversaciones y las respuestas estarán basadas en el contexto de las preguntas y respuestas anteriores.

### 7.1. Relación del tema con el sistema educativo actual

Este tema puede ser desarrollado en los siguientes módulos formativos (para atribución docente de PES)

- Bachillerato – Tecnologías de la Información y la Comunicación II (PES)
- GS- GS – DAW – Desarrollo Web en Entorno Servidor DAW/DAM – Programación (PES)

Aunque los profesores de SAI no tienen una atribución docente en ningún módulo con aplicación directa del tema, siempre es probable que se diseñe algún pequeño programa en clase, para lo que recurrir a la notación de pseudocódigo o de ordinograma sería un recurso esclarecedor.

## 8. Bibliografía

- Introduction to Java Programming and Data Structures, Comprehensive Version. Y. Daniel Liang. Pearson, 11<sup>a</sup> edición. 2017.
- Java 9. Francisco Javier Moldes Teo. Anaya. 2017.
- Introducción a la computación. J, Glenn Brookshear, Pearson, 11<sup>º</sup> edición. 2012
- Fundamentos de programación. Algoritmos, estructuras de datos y objetos. Luis Joyanes Aguilar, McGraw-Hill, 4<sup>º</sup> edición. 2008.
- Langsam, Augenstein y Tanembaum: “Estructuras de Datos con C y C++”, Prentice-Hall 1997
- Prieto A., Lloris A. y Torres J.C.: Introducción a la Informática, 4<sup>a</sup> ed (2006) McGraw-Hill
- Lenguajes de programación, principios y práctica. 2<sup>a</sup> Ed (2004). Kenneth C.Louden
- [https://es.wikipedia.org/wiki/Lenguaje\\_de\\_programaci%C3%B3n](https://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) (2020)

