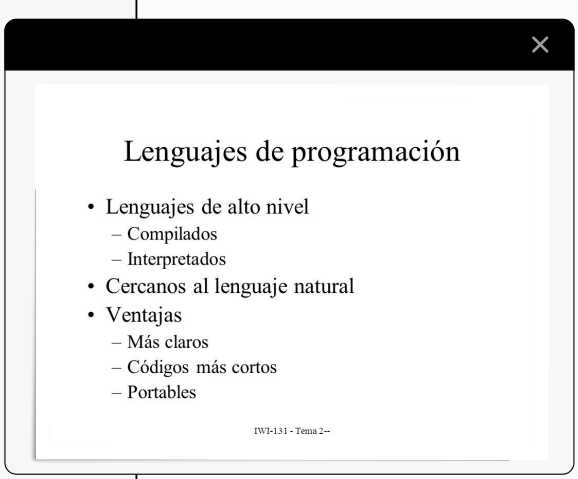
Inicio a la programación y sus fundamentos

Presentado por: Frannys Herrera

1. Lenguaje de programación

Lenguaje C
Lenguaje C++
Lenguaje Java
Lenguaje Python
Lenguaje Ruby
Lenguaje PHP
Lenguaje JavaScript
Lenguaje Swift
Lenguaje Kotlin
Lenguaje Objective-C

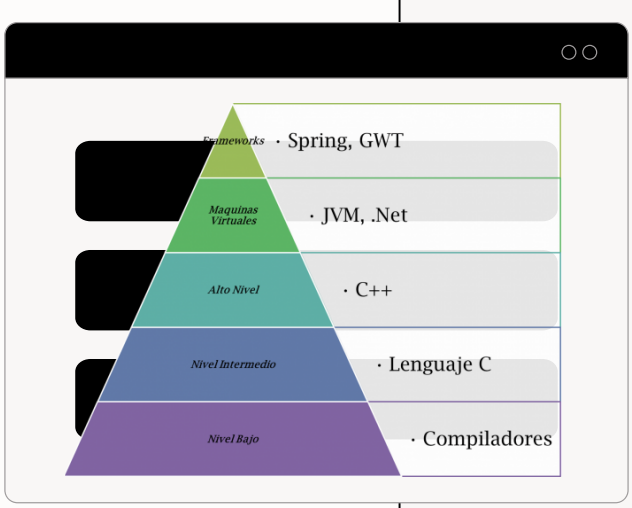


2. Lenguaje de alto nivel

Es un lenguaje de programación que se acerca más al lenguaje humano y permite al programador escribir instrucciones de manera comprensible y cercana a la forma en que pensamos y nos comunicamos. Estos lenguajes suelen ser más fáciles de aprender y utilizar, ya que se encargan de abstraer ciertos detalles técnicos del hardware, como la gestión de memoria, permitiendo al programador enfocarse en la lógica del programa. Se incluye C++, Java, Python.

3. Lenguaje de bajo nivel

Es un lenguaje de programación que se encuentra más cercano al lenguaje máquina y al hardware de la computadora. Estos lenguajes requieren que el programador tenga un mayor conocimiento técnico y detalle sobre el funcionamiento interno de la máquina, incluyendo la gestión de memoria, registros del procesador, y otros aspectos técnicos. Incluyen el lenguaje ensamblador y el lenguaje máquina.

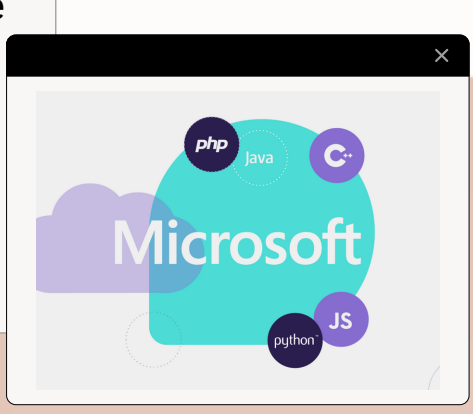


4. Proceso de copilación

Es el proceso mediante el cual un programa escrito en un lenguaje de alto nivel, como C++, se traduce a un lenguaje de bajo nivel, como lenguaje máquina, que la computadora puede entender y ejecutar. El compilador es el software encargado de realizar esta traducción, y durante el proceso de compilación se realizan diversas etapas como análisis léxico, análisis sintáctico, generación de código intermedio, optimización de código y generación de código objeto. Una vez completado el proceso de compilación, se obtiene un archivo ejecutable que puede ser ejecutado en la computadora.

5. Código de fuente

Es el programa escrito por el programador en un lenguaje de programación de alto nivel, como C. Este código fuente es legible para los humanos y contiene las instrucciones y algoritmos necesarios para que el programa realice sus funciones. El código fuente debe ser traducido a lenguaje de máquina mediante un compilador para que la computadora pueda ejecutarlo.

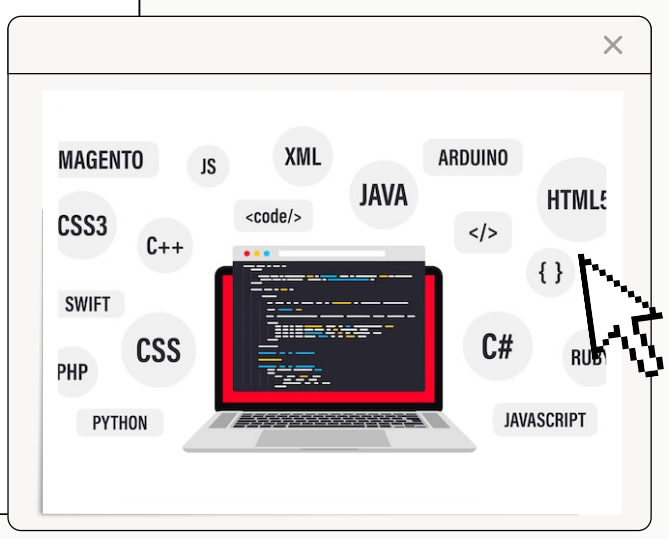


Fundamentos de la programación

Presentado por: Frannys Herrera

6. Código objeto

Es el resultado de la compilación del código fuente, pero aún no es un programa ejecutable. El código objeto contiene instrucciones en lenguaje de máquina que representan las operaciones del programa, pero todavía necesita ser enlazado con otras partes del programa para formar un ejecutable completo.



7. Código ejecutable

Es el resultado final del proceso de compilación, en el que el código objeto se combina con otras partes del programa (bibliotecas, módulos, etc.) para formar un archivo ejecutable. Este archivo puede ser ejecutado directamente por el sistema operativo.

8. Código de depuración

Es el proceso de identificar y corregir errores en el código fuente o en el programa ejecutable. La depuración puede realizarse mediante el uso de herramientas especiales (debuggers) que permiten examinar el estado interno del programa durante su ejecución, identificar problemas y corregirlos. También se puede realizar depuración mediante la inserción de mensajes de registro (logging) en el código para entender su comportamiento durante la ejecución.

9. Detección de errores

Consiste en identificar los errores en el código fuente o en el programa ejecutable. Estos errores pueden ser de sintaxis, lógica, o de otro tipo, y pueden causar que el programa no funcione correctamente o incluso falle. La detección de errores puede realizarse mediante el uso de herramientas de análisis estático o dinámico, pruebas unitarias, revisiones de código, entre otros métodos.

10. Corrección de errores

La corrección de errores puede implicar modificar el código fuente, ajustar la lógica del programa, o realizar cambios en el proceso de compilación. Es importante realizar pruebas exhaustivas después de corregir los errores para asegurarse de que el programa funciona correctamente.

11. Sintaxis de lenguaje C

La sintaxis de un lenguaje de programación, como el lenguaje C, se refiere a las reglas y estructuras que se deben seguir al escribir código en ese lenguaje. Esto incluye la forma en que se escriben las instrucciones, la manera en que se declaran las variables, la estructura de los bucles y condicionales, entre otros aspectos.

En el caso del lenguaje C, su sintaxis tiene reglas como el uso de punto y coma al final de cada instrucción, la declaración de variables antes de su uso, la estructura de los bucles con la palabra clave "for" o "while", y la utilización de llaves para delimitar bloques de código.

12. Semántica del lenguaje C

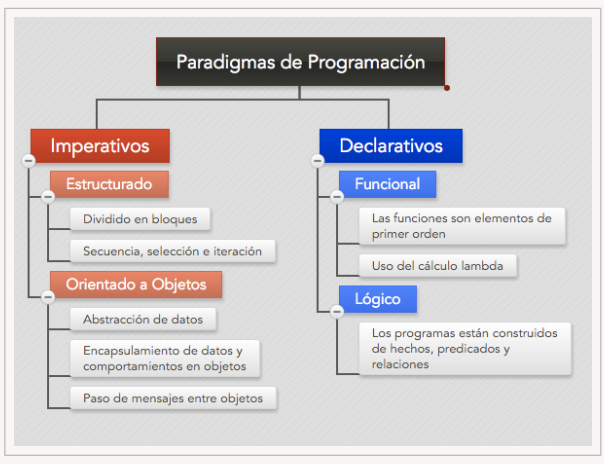
Se refiere al significado o interpretación de las instrucciones escritas en ese lenguaje. En el caso del lenguaje C, la semántica se refiere a cómo se ejecutan las instrucciones, cómo se manejan los tipos de datos, cómo se realizan las operaciones aritméticas y lógicas, y cómo se gestionan la memoria y los punteros.

13. Qué es un paradigma de programación

Un paradigma de programación es un enfoque o estilo particular para escribir programas de computadora. Cada paradigma tiene sus propias reglas, técnicas y filosofías que guían la forma en que se estructura y se escribe el código.

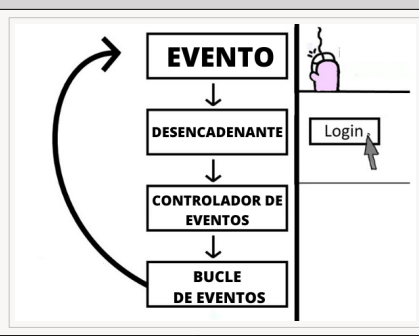
14. Ejemplos de paradigmas

- Programación imperativa: Se centra en describir cómo realizar una tarea.
- Programación orientada a objetos: Se estructuran entorno a objetos representados en el mundo real.
- Programación Funcional: Utiliza funciones matemáticas puras evitando cambios y efectos secundarios.
- Programación lógica: Se basa en la lógica matemática y la resolución de problemas a través de reglas y relaciones lógicas.



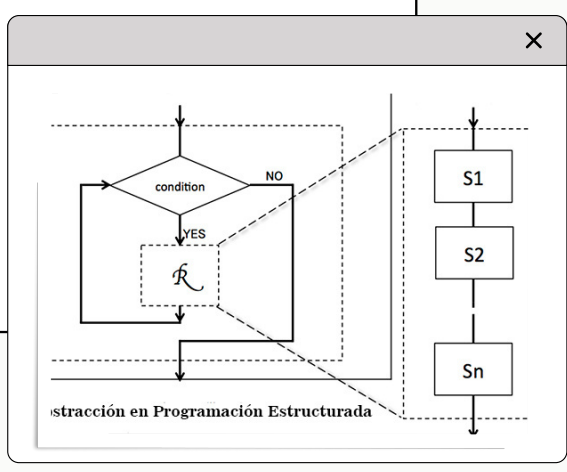
15. Programación secuencial

La programación secuencial es un enfoque en el que las instrucciones se ejecutan una tras otra, en secuencia. Cada instrucción se ejecuta en el orden en que aparece en el código, sin saltos ni bifurcaciones. Este enfoque es común en la programación imperativa, donde las instrucciones se utilizan para modificar el estado del programa y realizar tareas paso a paso.



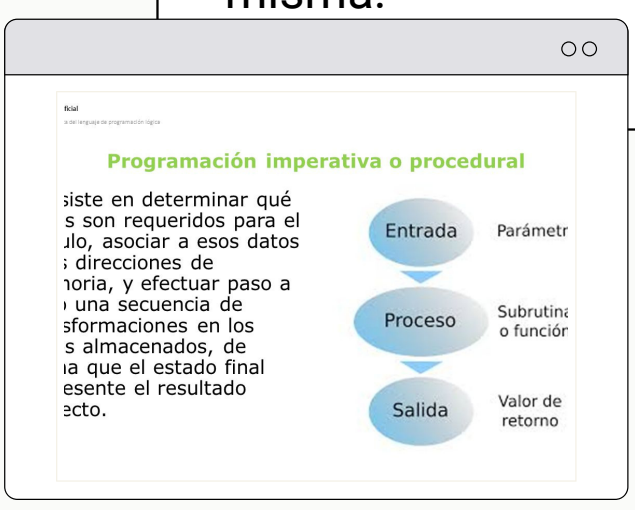
16. Programación estructurada

La programación estructurada es un paradigma de programación que se basa en la idea de dividir un programa en estructuras lógicas más pequeñas y más manejables. Se enfoca en el uso de instrucciones secuenciales, condicionales e iterativas para resolver problemas de manera eficiente.



17. Programación modular

En la programación recursiva, una función se divide en subproblemas más pequeños y se resuelve llamándose a sí misma con esos subproblemas. Cada llamada recursiva se acerca a la solución final del problema. La recursión se detiene cuando se alcanza una condición base o de salida que indica que ya no es necesario continuar llamándose a sí misma.



18. Programación recursiva

La programación modular es un enfoque que consiste en dividir un programa en módulos o unidades más pequeñas y autónomas. Cada módulo se encarga de una tarea específica y se comunica con otros módulos a través de interfaces bien definidas.