

A1.1 Conceptos iniciales

Investiga y responde a las siguientes preguntas/conceptos:

- 1-Qué es un *IDE* y di al menos 3 de los más usados.
- 2-Ciclo de vida de un software.
- 3-Fases en el desarrollo y ejecución del software.
- 4-Diferencia entre código fuente, código objeto y código ejecutable.
- 5-Diferencia entre algoritmo, pseudocódigo y código.
- 6-¿Qué es el *DISEÑO ESTRUCTURADO* y cuáles son los 3 tipos de construcciones en las que se basa?
- 7-Qué es un lenguaje de programación y clasificación de los lenguajes en función de su nivel de abstracción y según la forma de ejecución. Pon ejemplos.
- 8-Diferencia entre compilador e intérprete.
- 9-¿QUÉ ES BYTECODE?
- 10-En la programación orientada a objetos (POO), existen tres conceptos fundamentales: clase, objeto y método. Defínelos y pon un ejemplo práctico en un lenguaje que conozcas.

1-Un **IDE** (Entorno de Desarrollo Integrado, por sus siglas en inglés: **Integrated Development Environment**) es un **programa que reúne en una sola aplicación todas las herramientas necesarias para desarrollar software**

Visual Studio Code (VS Code) – Muy popular, gratuito y compatible con muchos lenguajes (JavaScript, Python, C++, etc.).

IntelliJ IDEA – Muy usado para Java y también para Kotlin y otros lenguajes.

Eclipse – Otro IDE clásico para Java, aunque también admite otros lenguajes.

2-El ciclo de vida del software es el conjunto de etapas que sigue un software desde que se idea hasta que deja de usarse. Sirve para organizar y planificar el desarrollo, mantenimiento y retiro del programa

Etapas del ciclo de vida del software

Etapas del ciclo de vida del software

1. **Análisis de requisitos**-Se recopila información sobre lo que el cliente o usuario necesita.
 - Se definen las funciones, características y objetivos del sistema.
2. **Diseño del sistema**-Se planifica **cómo** se va a construir el software.
 - Se define la arquitectura, la base de datos, la interfaz y los módulos del sistema.
3. **Implementación (codificación)**-Los programadores **escriben el código fuente** según el diseño realizado.
 - Se utilizan lenguajes de programación y herramientas de desarrollo (como un IDE).
4. **Pruebas (testing)**-Se **verifica** que el software funcione correctamente.
Se buscan y corrigen errores o fallos antes de la entrega al cliente.
5. **Implantación (despliegue)**-El software se **instala o publica** para que los usuarios puedan usarlo.
Puede incluir capacitación y documentación.
6. **Mantenimiento**-Se **corrigen errores**, se agregan nuevas funciones o se hacen mejoras.
 - Esta fase puede durar años, hasta que el software se reemplaza o deja de usarse

3-Fases del desarrollo y ejecución del software

Análisis: se estudian las necesidades. **Diseño:** se planifica cómo será el sistema. **Codificación:** se programa el software. **Pruebas:** se buscan y corrigen errores. **Ejecución:** se instala y se usa. **Mantenimiento:** se actualiza y mejora.

4-Diferencia entre código fuente, código objeto y código ejecutable.

1. Código fuente:

- *Es el código que escribe el programador en un lenguaje de programación (como Python, Java, C++, etc.).*
- *Es legible por humanos.*
- *Ejemplo:*
`printf("Hola mundo");`

2. Código objeto:

- *Es el resultado de compilar el código fuente.*
Está en lenguaje máquina parcial (no completamente ejecutable aún).
- *Es legible por la computadora, pero no por humanos.*
- *Suele tener extensión .obj o .o.*

3. Código ejecutable:

- *Es el programa final listo para ejecutarse en el sistema operativo.*
Se obtiene al enlazar (linkar) uno o varios códigos objeto.
- *Extensión común: .exe (en Windows).*
- *Es completamente en lenguaje máquina.*

5-Diferencia entre algoritmo, pseudocódigo y código.

ALGORITMO

- *Es una serie de pasos lógicos para resolver un problema.*
- *Se expresa en lenguaje natural (palabras comunes).*
- *No sigue una sintaxis de programación.*
- *Ejemplo: Paso 1: Leer dos números.
Paso 2: Sumarlos.
Paso 3: Mostrar el resultado.*

Pseudocódigo

- *Es una forma intermedia entre el algoritmo y el código.*
- *Usa una estructura parecida a un lenguaje de programación, pero sin reglas estrictas.*
- *Sirve para planificar el programa antes de escribirlo en código real.*

Leer A, B - Suma ← A + B - Escribir Suma

6-¿Qué es el DISEÑO ESTRUCTURADO y cuáles son los 3 tipos de construcciones en las que se basa?

Método para crear programas claros y organizados dividiéndolos en partes pequeñas.

Se basa en tres estructuras:

1. **Secuencia: pasos en orden.**
2. **Selección: decisiones (si / sino).**
3. **Iteración: repeticiones (bucles).**

7-Qué es un lenguaje de programación y clasificación de los lenguajes en función de su nivel de abstracción y según la forma de ejecución. Pon ejemplos

Es un conjunto de reglas y símbolos que permite dar instrucciones a una computadora para crear programas o aplicaciones. Sirve para comunicar al programador

Según su nivel de abstracción:

- **Bajo nivel: muy cercano al lenguaje máquina, difícil de leer.**
Ejemplo: Assembly (ASM)
- **Medio nivel: combina control del hardware con facilidad de uso.**
Ejemplo: C
- **Alto nivel: fácil de entender, cercano al lenguaje humano.**
Ejemplos: Python, Java, C++, JavaScript, PHP

Según la forma de ejecución:

- **Compilados: se traducen completamente a lenguaje máquina antes de ejecutarse.**
Ejemplos: C, C++, Rust, Go
- **Interpretados: se ejecutan línea por línea mediante un intérprete.**
Ejemplos: Python, JavaScript, PHP, Ruby
- **Híbridos: se compilan parcialmente y luego se interpretan.**
Ejemplos: Java, C#

8-Diferencia entre compilador e intérprete

Compilador

- **Traduce todo el programa completo a lenguaje máquina antes de ejecutarlo.**
- **Genera un archivo ejecutable (.exe, .out, etc.).**
- **La ejecución es más rápida.**
- **Si hay errores, se muestran después de compilar todo.**

Intérprete

- Traduce y ejecuta el programa línea por línea, mientras se ejecuta.
- No genera un archivo ejecutable.
- La ejecución es más lenta, pero más flexible para probar código.
- Los errores se muestran al momento de ejecutarse.

9-¿QUÉ ES BYTECODE?

El bytecode (código intermedio) es un tipo de código que no está ni en lenguaje humano ni en lenguaje máquina, sino en un punto medio.

Se genera cuando un lenguaje híbrido, como Java o Python, compila el código fuente. Este bytecode no puede ejecutarse directamente por el procesador, sino que necesita una máquina virtual que lo interprete (por ejemplo, la Java Virtual Machine – JVM).

Proceso general:

1. El programador escribe el código fuente.
2. El compilador lo convierte en bytecode.
3. Una máquina virtual (JVM, PVM, etc.) lo interpreta y lo ejecuta en cualquier sistema.

10-En la programación orientada a objetos (POO), existen tres conceptos fundamentales: clase, objeto y método. Defínelos y pon un ejemplo práctico en un lenguaje que conozcas

Conceptos fundamentales de la Programación Orientada a Objetos (POO)

1. Clase:

Es un modelo o plantilla que define cómo serán los objetos. Describe atributos (datos) y métodos (acciones). Ejemplo: una clase Coche define que todos los coches tienen marca, modelo y pueden acelerar o frenar.

2. Objeto:

Es una instancia (ejemplo real) de una clase.

Representa un elemento concreto con sus propios valores. Ejemplo: un objeto puede ser miCoche = Coche("Toyota", "Corolla").

3. Método: Es una función dentro de una clase que define el comportamiento del objeto. Ejemplo: un método acelerar() o frenar() dentro de la clase Coche.

