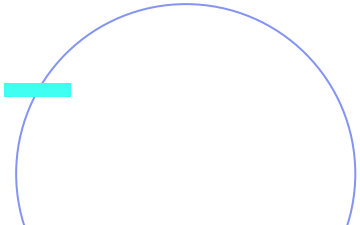
Glosario de términos

- **Administradores de Paquetes:** Herramientas que permiten instalar, actualizar y gestionar librerías y dependencias. Aseguran la compatibilidad y que la versión de las dependencias sea la correcta.
- **Agregación:** Tipo especial de asociación donde un objeto contiene a otros, pero los objetos contenidos pueden existir por sí mismos.
- **Atributo:** Son variables que pertenecen a una clase y describen el estado de un objeto. Especifican las propiedades que distinguen a cada objeto creado a partir de esa clase, reflejando sus características individuales. También se los conoce como variables de instancia.
- **Asociación:** Relación entre objetos que establece una dependencia entre ellos, pero sin que uno forme parte del otro.



- ***Backend:*** Es la parte del *software* que maneja la lógica de negocio, bases de datos y procesamiento del servidor. No es visible para el usuario y se comunica con el *frontend* para entregar funcionalidades.
- **Clase:** Plantilla que define las propiedades y métodos comunes a todos los objetos de un cierto tipo.
- ***Coding Assistants:*** Herramientas basadas en IA que ayudan a los desarrolladores a escribir, depurar y optimizar código. Ofrecen sugerencias, autocompletado, corrección de errores y documentación en tiempo real.
- **Composición:** Relación entre objetos en la que uno es parte del otro, pero pueden existir independientemente.
- **Constructor:** Método especial de una clase que se invoca automáticamente cuando se crea una nueva instancia de esa clase. Su función principal es inicializar los atributos del objeto, asignando valores iniciales o configurando estados.

- **Control de excepciones:** Técnica para manejar errores y eventos inesperados que ocurren durante la ejecución de un programa. La mayoría de los lenguajes de programación orientados a objetos proporcionan estructuras para manejar excepciones. Los elementos más comunes son: `try`, `catch`, `finally`.
- **Dependencia:** Relación entre clases donde una clase utiliza otra para realizar una tarea.
- **Encapsulamiento:** Técnica de ocultar los detalles internos de un objeto y exponer sólo lo necesario a través de métodos públicos. Esto protege la integridad de los datos y evita accesos no controlados o modificaciones indebidas.
- **Excepciones:** Eventos o condiciones anómalas que interrumpen el flujo normal de ejecución de un programa. Representan situaciones inesperadas que requieren un manejo especial. Por ejemplo: intentar acceder a un objeto que no ha sido inicializado o a una posición inválida de un vector.
- ***Frameworks:*** Plataformas de desarrollo que proporcionan estructuras y herramientas predefinidas para crear aplicaciones. Facilitan la organización del código, reutilización de componentes y garantizan consistencia en proyectos.

- ***Frontend:*** Parte visual de una aplicación que interactúa directamente con el usuario. Incluye el diseño, la interfaz y la experiencia de usuario, ejecutándose en el navegador mediante HTML, CSS y JavaScript.
- **Git:** Es un sistema de control de versiones distribuido. Permite que cada desarrollador tenga una copia completa del historial del proyecto en su computadora. Esto permite trabajar de forma más independiente y eficiente.
- **GitHub:** Plataforma de desarrollo colaborativo que utiliza Git para almacenar y administrar proyectos. Es como un repositorio en la nube donde puedes alojar tus proyectos de Git y colaborar con otros desarrolladores.
- **Herencia:** Mecanismo por el cual una clase (subclase) puede heredar propiedades y métodos de otra clase (superclase).
- **Instancia:** Objeto específico creado a partir de una clase.

- **Interfaz:** Contrato que define un conjunto de métodos abstractos que una clase debe implementar. No contiene implementación, solo la firma de los métodos. Sirve como plantilla para crear clases relacionadas sin establecer una jerarquía de herencia directa. Promueve el polimorfismo y la desacoplamiento entre clases.

- ***Large Language Models:*** Algoritmos de inteligencia artificial entrenados con enormes cantidades de texto para comprender y generar lenguaje humano. Utilizan redes neuronales profundas, especialmente transformadores, para realizar tareas como traducción, resumen, respuesta a preguntas y generación de texto coherente.

- **Librerías:** Son colecciones de funciones y recursos reutilizables que facilitan el desarrollo de *software* permitiendo a los desarrolladores reutilizar bloques de código para ahorrar tiempo y simplificar la implementación de funcionalidades específicas.
- **Mensaje:** Comunicación entre objetos mediante la invocación de métodos.

- **Mermaid:** Herramienta que permite crear diagramas y gráficos con una sintaxis de texto simple. Es ideal para documentar flujos de trabajo, organigramas, diagramas de clases, diagramas de Gantt, entre otros, directamente en el código o en archivos de documentación. Es popular en plataformas como *GitHub* y *Markdown.*
- **Método:** Función definida dentro de una clase que describe el comportamiento de los objetos de esa clase.
- **Módulo:** Unidad de código independiente que agrupa clases y funciones relacionadas.

- **Objeto:** Una instancia de una clase. Es un ente concreto que posee estado (datos) y comportamiento (métodos).
- **Polimorfismo:** Capacidad de los objetos de diferentes clases de responder a la misma operación o método de diferentes maneras.
- ***Prompt Engineering:*** Técnica de diseñar y ajustar las instrucciones *(prompts)* que se le dan a un modelo de lenguaje, como ChatGPT o Copilot, para obtener resultados específicos y de alta calidad. Implica elegir las palabras correctas, estructurar la pregunta de manera clara y concisa, y proporcionar contexto para obtener respuestas relevantes.
- **Propiedades:** Aspectos de un objeto que describen su estado, generalmente implementados como variables de instancia.
- **Pruebas unitarias:** Son pruebas automatizadas que verifican la funcionalidad de unidades individuales de código, como funciones o métodos. Su objetivo es detectar errores en una etapa temprana del desarrollo, asegurando que cada unidad de código funcione como se espera.

- **Sobrecarga de constructores:** permite a una clase tener múltiples constructores con diferentes listas de parámetros, proporcionando diversas formas de instanciar objetos.
- **Sobrecarga de métodos:** Capacidad de definir múltiples métodos con el mismo nombre pero con diferentes parámetros en una clase.
- **Sobrescritura de métodos:** Capacidad de una subclase de redefinir un método de su superclase para cambiar o extender su comportamiento.
- ***Unified Modeling Language (UML):*** Lenguaje estándar para visualizar, especificar, construir y documentar los artefactos de un sistema de software. UML utiliza una notación gráfica para representar los elementos de un sistema, como clases, objetos, relaciones y comportamientos. Sirve como un lenguaje común para que los desarrolladores, analistas y otros interesados se comuniquen y colaboren en el diseño y desarrollo de *software.*

Ahora si,
¡comencemos!