FRAGUA TECH

Desarrollo de Software Moderno con IA

Clase 1 — Introducción al Desarrollo de Software Moderno

↓

¿Por qué este curso?

Estamos viviendo el **auge de la Inteligencia Artificial**. Los modelos de lenguaje ya son capaces de escribir, refactorizar y explicar código.

Esto ha dado lugar a **nuevas formas de desarrollar software** que están transformando la industria a una velocidad sin precedentes.

El desarrollo de software ya no se trata solo de escribir código: se trata de **saber dirigir, revisar y comprender** lo que las herramientas generan.

Tres tipos de desarrolladores

Vibe Coders

Generan software casi enteramente con IA, sin profundizar en el código.

En aumento

Asistidos por IA

Usan la IA como copiloto, pero entienden, revisan y dirigen el código.

Recomendado

Tradicionales

Escriben todo manualmente, sin asistencia de modelos de lenguaje.

En declive

Vibe Coders

El **vibe coding** consiste en describir lo que se quiere construir y dejar que la IA genere todo el código, sin necesariamente entender cómo funciona.

Este enfoque está en **fuerte aumento** gracias a la potencia de los modelos de lenguaje actuales: GPT-4, Claude, Gemini, etc.

Permite crear prototipos y MVPs a una velocidad impresionante, lo que lo hace muy atractivo para emprendedores y no-programadores.

Programadores asistidos por IA

Utilizan la IA como un **copiloto inteligente**: generan código, pero lo revisan, comprenden y modifican cuando es necesario.

Combinan el poder de la automatización con el **criterio técnico** para tomar decisiones de arquitectura, depuración y escalabilidad.

Este perfil es el que más demanda tiene en la industria actual y el que este curso busca formar.

Desarrolladores tradicionales

El desarrollador que escribe todo manualmente, línea por línea, sin asistencia de modelos de lenguaje.

Este rol se encuentra en **declive**. No porque los conocimientos técnicos pierdan valor, sino porque la velocidad y productividad que ofrece la IA hacen que este enfoque sea menos competitivo.

Los conocimientos fundamentales siguen siendo esenciales, pero la forma de aplicarlos está cambiando.

Vibe Coding: alcance y límites

El vibe coding tiene un **gran alcance** para prototipos, scripts y proyectos pequeños. Pero **no es la solución a todo**.

Los modelos de lenguaje suelen fallar cuando se trata de desarrollar **features complejas** que requieren un entendimiento profundo del contexto del proyecto.

A medida que el software crece, los **errores se acumulan** y se vuelven más difíciles de resolver sin comprensión real del código.

Esto hace que el vibe coding **no sea viable para proyectos con intención de escalar**.

Un nuevo modelo de aprendizaje

Los cambios en la industria exigen un **cambio en cómo aprendemos**.

Cada vez es más importante **entender y abstraerse** por encima de memorizar sintaxis o APIs.

La **creatividad** se vuelve más necesaria que la acumulación de conocimientos puntuales — aunque los conocimientos fundamentales **siguen siendo muy importantes**.

Ahora más que nunca, es esencial dominar la **lectura de código** y la **depuración**: saber leer lo que la IA genera y encontrar errores rápidamente.

La programación asistida es el camino

Con las herramientas actuales, **crear software es más rápido que nunca**. Pero también lo es acumular deuda técnica.

La buena noticia: con IA también es mucho más rápido **identificar y eliminar esa deuda técnica**.

Por eso la **programación asistida por IA** es la mejor opción: combina velocidad de desarrollo con la capacidad de mantener el código limpio, entendible y escalable.

Antes de programar: conceptos fundamentales

Antes de adentrarnos en el desarrollo de software, necesitamos entender los pilares sobre los que se construye toda la informática.

Memoria

Cómo los programas almacenan y acceden a datos, tanto de forma permanente como temporal.

Redes

Cómo las computadoras se comunican entre sí a través de protocolos como TCP/IP.

Algoritmia

Cómo medir la eficiencia de un programa y estimar cuánto tardará en ejecutarse.

Manejo de memoria

Memoria estática (ROM / Disco)

Almacenamiento permanente: discos duros, SSDs. Los datos persisten aunque se apague el equipo.

Memoria dinámica (RAM / VRAM)

Almacenamiento temporal y ultra-rápido: RAM, tarjetas de video. Se borra al apagar el equipo.

Memoria estática — Disco

Los programas almacenan sus datos de forma permanente en el disco mediante **bases de datos, ficheros, imágenes, configuraciones**, etc.

Cuando apagas el computador y lo vuelves a encender, todo lo que estaba en disco **sigue ahí**. Por eso se llama memoria persistente.

Es más lenta que la RAM, pero tiene mucha más capacidad y no es volátil.

⚡

Memoria dinámica — RAM

La RAM es donde **se ejecutan los programas**. Cuando abres una aplicación, sus datos se cargan del disco a la RAM para que la CPU pueda procesarlos rápidamente.

Es **volátil**: al apagar el equipo, todo lo que estaba en RAM desaparece.

Las tarjetas de video (GPU) tienen su propia memoria dinámica — la **VRAM** — optimizada para cálculos gráficos y de IA.

📚 Material complementario ▾

Redes y TCP/IP

Toda comunicación en internet se basa en un modelo de capas. Cada capa tiene una responsabilidad específica.

Las más relevantes para el desarrollo de software son:

Capa 3 — Red

IP: direccionamiento y enrutamiento

Capa 4 — Transporte

TCP y UDP: entrega de datos

Capa 7 — Aplicación

HTTP: la web tal como la conocemos

Capa 3 — Red (IP)

El protocolo **IP (Internet Protocol)** se encarga de asignar direcciones únicas a cada dispositivo en la red y de encontrar la ruta para que los datos lleguen a su destino.

Cada dispositivo conectado tiene una **dirección IP** — como una dirección postal para los paquetes de datos.

IPv4 usa direcciones como `192.168.1.1`, mientras que IPv6 amplía enormemente el espacio disponible.

Capa 4 — Transporte (TCP / UDP)

TCP (Transmission Control Protocol) garantiza que los datos lleguen completos y en orden. Es el protocolo detrás de la web, el email y las transferencias de archivos.

UDP (User Datagram Protocol) es más rápido pero no garantiza entrega ni orden. Se usa en videollamadas, streaming y videojuegos, donde la velocidad importa más que la perfección.

Los **puertos** permiten que un mismo dispositivo maneje múltiples conexiones simultáneas (ej: puerto 80 para HTTP, 443 para HTTPS).

Capa 7 — Aplicación (HTTP)

HTTP (HyperText Transfer Protocol) es el lenguaje que hablan los navegadores y los servidores web.

Funciona con un modelo de **petición y respuesta**: el cliente pide un recurso (una página, una API) y el servidor responde con datos y un código de estado (200 OK, 404 Not Found, 500 Error...).

HTTPS añade cifrado TLS sobre HTTP, protegiendo los datos en tránsito. Hoy en día es el estándar.

Como desarrolladores web, **viviremos en esta capa**: APIs REST, WebSockets, headers, cookies — todo pasa aquí.

📚 Material complementario ▾

CPU y complejidad algorítmica

La **CPU** es el cerebro del computador: ejecuta las instrucciones de los programas una por una, a velocidades de miles de millones de operaciones por segundo.

Conocer la **complejidad algorítmica** nos permite estimar cuánto tardará un programa en función del tamaño de los datos.

Se expresa con la notación **Big O**:

O(1)	O(log n)	O(n)	$O(n^2)$
Constante	Logarítmica	Lineal	Cuadrática

Un algoritmo **O(1)** tarda lo mismo sin importar los datos. Uno **$O(n^2)$** se vuelve exponencialmente más lento con más datos — la diferencia entre milisegundos y horas.

📚 **Material complementario** ▾

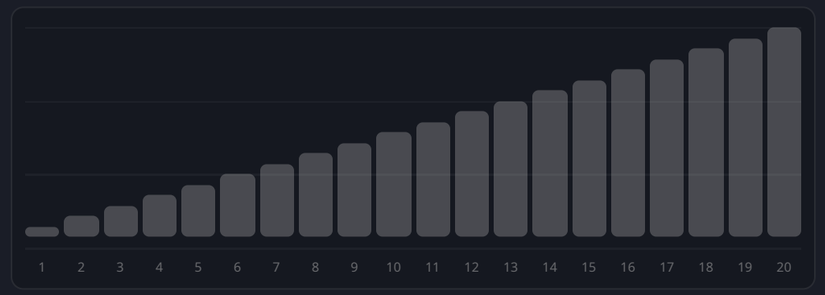
Búsqueda lineal vs binaria

La búsqueda lineal revisa cada elemento uno por uno **O(n)**. La binaria descarta la mitad en cada paso **O(log n)**. Observa la diferencia.

Buscando: **1** Iniciar búsqueda

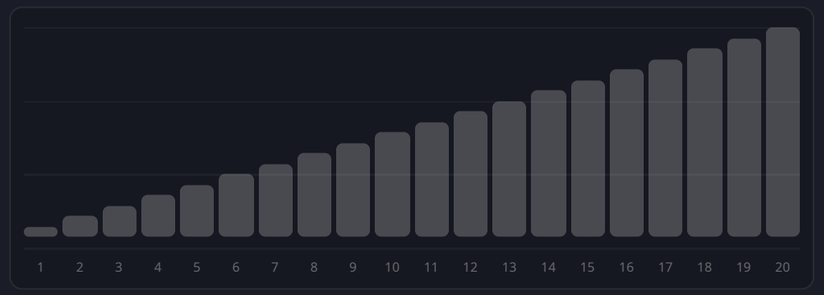
Búsqueda lineal — O(n)

Revisa uno por uno



Búsqueda binaria — O(log n)

Divide y conquista



Sin revisar · En rango · Revisando · Encontrado

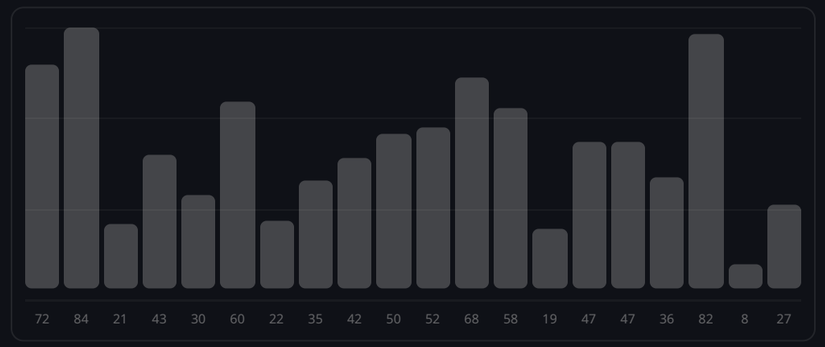
Burbuja vs Merge Sort

Burbuja compara pares adyacentes **O(n²)**. Merge Sort divide, ordena y combina **O(n log n)**. Con 20 elementos la diferencia ya es visible.

Iniciar ordenamiento

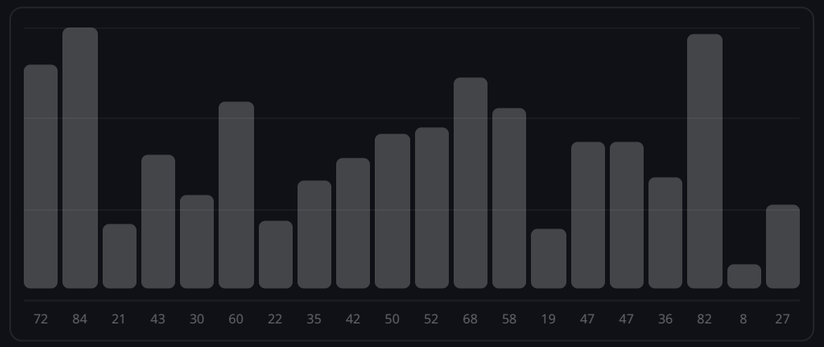
Burbuja — O(n²)

Compara pares adyacentes



Merge Sort — O(n log n)

Divide, ordena y combina



Sin ordenar · Comparando · Evaluando · Ordenado

Más que un curso: una comunidad

El objetivo de este curso va más allá de compartir conocimientos técnicos. Queremos **construir una comunidad** de personas que aprenden, crean y crecen juntas.

Una comunidad que **crezca con el tiempo**, donde cada persona que aprende algo nuevo lo comparta con los demás, generando un ciclo de aprendizaje continuo.

Lo más importante es que no se pierda este **espíritu de compartir**. El conocimiento tiene más valor cuando se multiplica, no cuando se acumula.

Bienvenidos a la Fragua

Esta será la fragua donde se forjarán los **desarrolladores del mañana**.

Fragua Tech — Clase 1