

Temario Unificado de Programación en Java (con explicaciones breves)

Módulo 0: Introducción y Preparación

- **¿Qué es Java y por qué elegirlo?:** Contexto histórico y aplicaciones de Java (móvil, web, empresarial).
- **JVM, JRE y JDK:** Diferencias entre el motor de ejecución, el entorno de ejecución y el kit de desarrollo.
- **Instalación y configuración:** Instalar el JDK y un IDE como IntelliJ, VSCode o NetBeans.
- **Primer programa ("Hola Mundo"):** Crear y ejecutar el programa básico.

Módulo 1: Fundamentos de Programación

- **Sintaxis básica de Java:** Estructura de clases y main.
- **Variables y tipos de datos:** Primitivos, Strings y conversiones.
- **Palabras reservadas y alcance:** Uso correcto de palabras clave y visibilidad de variables.
- **Operadores:** Aritméticos, lógicos, relacionales y de asignación.
- **Errores y convenciones (lógicos y sintácticos):** Buenas prácticas, errores comunes y comentarios.
- **Entrada/salida:** System.out.println, Scanner.

✅ **Práctica:** Calculadora en consola.

Módulo 2: Control de Flujo y Métodos

- **Estructuras condicionales:** if, else, switch, anidados.
- **Bucles:** for, while, do-while.
- **Funciones y métodos:** Parámetros, retorno, métodos estáticos y sobrecarga.
- **Recursividad:** Métodos que se llaman a sí mismos (factorial, Fibonacci).

✅ **Práctica:** Tablas de multiplicar, FizzBuzz, factorial recursivo.

Módulo 3: Arreglos y Colecciones

- **Arreglos:** Unidimensionales y multidimensionales, operaciones básicas.
- **Colecciones básicas:** ArrayList, LinkedList, HashMap, HashSet.

- **Operaciones comunes:** búsqueda, ordenamiento simple.

✓ **Práctica:** Mini-sistema de gestión de estudiantes.

Módulo 4: Programación Orientada a Objetos (POO)

- **Abstracción:** Definir entidades a partir de problemas reales.
- **Clases y objetos:** Crear y usar instancias.
- **Encapsulamiento:** Uso de modificadores de acceso, getters y setters.
- **Herencia:** Reutilizar código con extends, super.
- **Polimorfismo:** Sobrecarga y sobreescritura de métodos.
- **Constructores y this.**
- **Interfaces y clases abstractas.**

✓ **Proyecto 1:** Sistema bancario simple .

Módulo 5: Manejo de Excepciones y Archivos

- **Excepciones:** try-catch-finally, throw/throws, excepciones personalizadas.
- **Archivos:** Lectura y escritura con FileReader, BufferedReader, FileWriter, Streams.

✓ **Práctica:** Guardar y leer historial de un sistema.

Módulo 6: Algoritmos y Estructuras Intermedias

- **Algoritmos de ordenamiento:** burbuja, quicksort, mergesort básico.
- **Búsqueda binaria.**
- **Estructuras intermedias:** Stack, Queue, Map.
- **Genéricos:** Plantillas de clases y métodos.

✓ **Proyecto 2:** Gestor de tareas en consola.

Módulo 7: Java Avanzado

- **Programación funcional:** Lambdas, funciones anónimas, interfaces funcionales.
- **Streams:** Procesamiento eficiente de colecciones.

- **Concurrencia:** Threads(Hilos), synchronized, wait/notify, Future, atomicidad.
- **Optional y API Collections.**
- **Maven, Gradle, Ant:** gestión de dependencias.

✓ **Práctica:** Procesar datos con Streams.

Módulo 8: Bases de Datos

- **SQL básico:** CREATE, INSERT, SELECT, UPDATE, DELETE.
- **Filtros y joins:** WHERE, ORDER BY, LIMIT, INNER/LEFT/RIGHT JOIN.
- **Conexión con JDBC.**
- **CRUD completo desde Java.**
- **Hibernate (ORM).**

✓ **Proyecto 3:** Sistema de inventario con base de datos.

Módulo 9: Interfaces Gráficas

- **Swing:** Ventanas, botones, eventos.
- **JavaFX:** Escenas, componentes, Scene Builder.

✓ **Proyecto 4:** Calculadora con GUI.

Módulo 10: Ecosistema y Frameworks

- **Jakarta EE:** Servlets, JSP, JSF, JPA, REST.
- **Spring Boot:** Fundamentos para aplicaciones modernas.
- **Quarkus** (extra).
- **Librerías útiles:** Lombok, Data Faker, iTextPDF.
- **Empaquetado JAR y módulos.**

Módulo 11: Control de Versiones

- **Git básico:** commits, ramas, merge, conflictos.
- **Colaboración en GitHub.**

- ✅ **Proyecto mini:** Versionar el sistema bancario del módulo 4.

Proyecto Final Integrador

Una **aplicación completa** con:

- POO, colecciones, algoritmos y excepciones
- Persistencia con BD (JDBC/ORM)
- Interfaz gráfica (Swing o JavaFX)
- Git y GitHub para control de versiones
- Opcional: integración con Spring Boot y Jakarta EE

Ejemplo: **Sistema de gestión bancaria y de tareas con GUI y BD.**

Resumen de aportes únicos:

- **Conceptos básicos (JVM/JRE/JDK, convenciones, errores comunes, paquetes, concurrencia avanzada, frameworks, librerías externas)** → vienen del **Temario de Alexander**.
- **Proyectos acumulativos y mini-prácticas claras (FizzBuzz, gestor de tareas, banco, inventario, Git)** → vienen del **Temario de Ian**.
- **Énfasis en modularidad y buenas prácticas desde el inicio** → vienen del **Temario de Fabrizio**.
- **Integración temprana con BD y GUI (Swing + JDBC)** → del **Temario de Diana**

HOJA DE EVIDENCIA DE FIRMA DIGITAL

Firmado por: Ian Villamil

Correo electrónico: ianvillamil06@gmail.com

Fecha de firma (UTC): 2025-11-18 07:19:59

Huella Digital (Hash SHA-256):

c5fbaf915bfc8085987880ed536acb00dafa3e3449e6ba2df66ce741b3a71ba1



HOJA DE EVIDENCIA DE FIRMA DIGITAL

Firmado por: Mauricio Clemente

Correo electrónico: mauricioclementebarragan@gmail.com

Fecha de firma (UTC): 2025-11-19 19:17:32

Huella Digital (Hash SHA-256):

af4c9a7e1dfff36fc0b50b4aa39d4780eacf4659df3b45f0f13f2fcb0d463f2ed

