Paradigmas de Programación — Plan de Clases (Cuatrimestre, 16 clases)

*Herramienta: Node.js (JavaScript) • Modalidad: 20% teoría / 80% práctica • 16 semanas (2 de examen)*

# Cronograma resumido (ver detalles en la tabla):

Clases 1–3: Fundamentos, HOF, map/filter/some/every

Clases 4–7: reduce, sort, fs/JSON, asincronía

Clase 8: Parcial 1

Clases 9–15: OO básica, módulos, pruebas, TDAs, generators, validaciones, CLI, clínica de proyecto

Clase 16: Parcial 2 / Examen integrador

|  |  |  |
| --- | --- | --- |
| Clase | Título | Enfoque y actividades clave |
| 1 | Introducción + entorno Node.js | Paradigmas; Node y npm; funciones puras; práctica de expresiones. |
| 2 | Orden superior, closures | Callbacks, arrow functions; generadores de funciones; ejercicios guiados. |
| 3 | map/filter/some/every | Pipelines legibles; katas de arrays; validaciones con every/some. |
| 4 | reduce + sort | Acumulaciones y agrupamientos; comparadores; evitar mutaciones. |
| 5 | Pipelines y composición | Currying; refactor imperativo→funcional; legibilidad. |
| 6 | Arrays de objetos, JSON y fs | readFile/writeFile; limpieza y dedup; mini-ETL. |
| 7 | Asincronía | Promises y async/await; agregación de resultados; manejo de errores. |
| 8 | Parcial 1 | Ejercicios prácticos individuales sobre colecciones y FP. |
| 9 | Comparación de paradigmas + OO | Clases/objetos; this; una solución en FP y otra en OO. |
| 10 | Módulos, npm scripts, pruebas | ESM, estructura; node:assert; scripts de build/test. |
| 11 | TDAs y algoritmos | Stack/Queue; complejidad; sorting y medición simple. |
| 12 | Iterables/Generators | Evaluación perezosa; pipelines eficientes. |
| 13 | Validaciones y errores | every/some para reglas; diseño de mensajes y resultados. |
| 14 | CLI profesional | process.argv; help; UX de terminal; ejemplos reproducibles. |
| 15 | Clínica + simulacro | Revisión por pares; defensa corta del proyecto. |
| 16 | Parcial 2 / Integrador | Ejercicios + defensa breve del proyecto. |