

Archimedes Indexer

De Python Básico a Pathway MS AI (CU Boulder)

DUQUEOM · 2025 · Versión 1.0

Índice Principal

Guía Archimedes Indexer

DUQUEOM · 2025



Archimedes Indexer - Índice Principal

De Python Básico a Ingeniero de IA: Construyendo un Motor de Búsqueda desde Cero



¿Qué Lograrás al Completar Esta Guía?

Habilidad	Nivel Alcanzado	Evidencia
OOP Profesional	Avanzado	Clases Document, Corpus, InvertedIndex con diseño SOLID
Estructuras de Datos	Intermedio-Avanzado	Hash Maps, Tries, Listas enlazadas implementadas desde cero
Algoritmos	Intermedio-Avanzado	QuickSort, Binary Search, recursión dominada
Álgebra Lineal Aplicada	Intermedio	TF-IDF y Similitud de Coseno sin numpy
Análisis de Complejidad	Intermedio	Documentación Big O de todo el sistema
Inglés Técnico	B2+	Todo el material y código en inglés



Perfil de Entrada

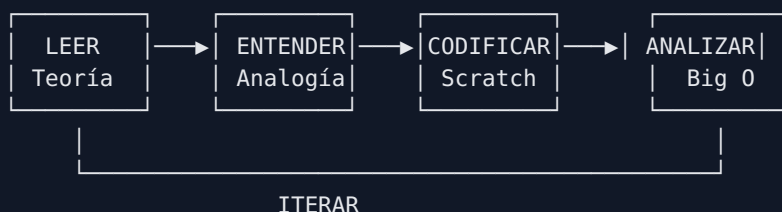
PERFIL IDEAL DE ENTRADA

- ✓ Python básico (variables, funciones, listas, diccionarios)
- ✓ Lógica de programación (if/else, loops)
- ✓ Ganas de entender "cómo funciona por dentro"
- ✓ Matemáticas de bachillerato (álgebra básica)
- ⚠ NO se requiere: numpy, pandas, sklearn, ML previo



Metodología de Aprendizaje

CICLO DE APRENDIZAJE



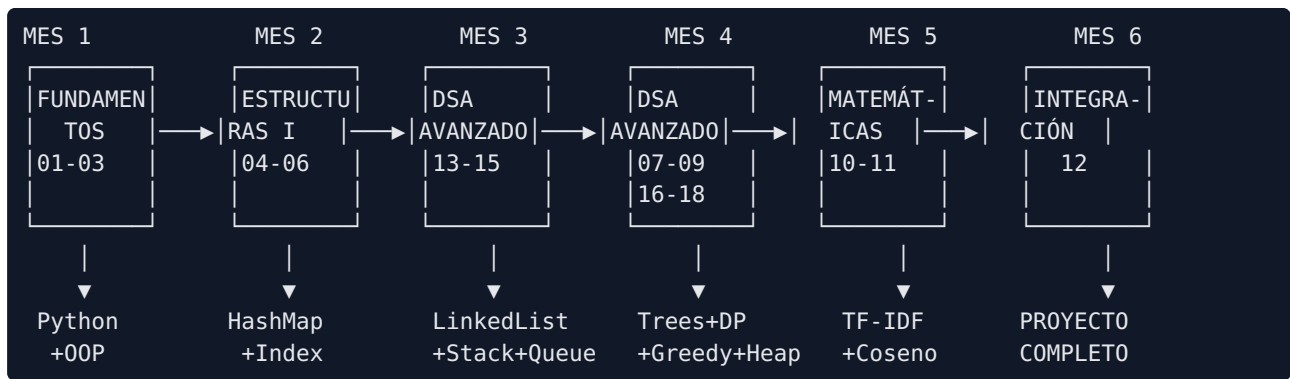
Leer: Estudiar el concepto teórico y su contexto.

Entender: Asimilar mediante analogías y ejemplos visuales.

Codificar: Implementar desde cero en Python puro.

Analizar: Documentar la complejidad y eficiencia.

Roadmap Visual (6 Meses)



Módulos de la Guía

FASE I: Fundamentos (Mes 1)

Objetivo: Establecer bases sólidas de Python profesional y pensamiento computacional

#	Módulo	Descripción	Tiempo
01	Python Profesional	Type hints, funciones puras, estilo PEP8	2 semanas
02	OOP desde Cero	Clases, herencia, composición, SOLID básico	1.5 semanas
03	Lógica y Matemáticas Discretas	Conjuntos, lógica proposicional, demostraciones	0.5 semanas

Checkpoint Fase I: [Simulacro Fundamentos](#)

FASE II: Estructuras de Datos (Mes 2-3)

Objetivo: Dominar las estructuras que hacen rápidos a los sistemas reales

#	Módulo	Descripción	Tiempo
04	Arrays, Strings y Memoria	Manipulación de secuencias, complejidad básica	1.5 semanas
05	Hash Maps y Sets	Diccionarios, hashing, colisiones, O(1) lookup	2 semanas
06	Índice Invertido	Construcción del núcleo del motor de búsqueda	2.5 semanas

Checkpoint Fase II: [Simulacro Estructuras](#)

FASE III: Estructuras de Datos Avanzadas (Mes 3) ★ CRÍTICO PATHWAY

Objetivo: Dominar estructuras que aparecen en el Pathway

#	Módulo	Descripción	Tiempo
13	Linked Lists, Stacks, Queues	Estructuras lineales fundamentales	1.5 semanas
14	Árboles y BST	Binary trees, traversals, BST	2 semanas
15	Grafos, BFS, DFS	Representación, recorridos	2 semanas

Checkpoint Fase III: [Simulacro DSA Avanzado](#)

FASE IV: Algoritmos (Mes 4) ★ CRÍTICO PATHWAY

Objetivo: Implementar algoritmos clásicos de ordenamiento, búsqueda y optimización

#	Módulo	Descripción	Tiempo
07	Recursión y Divide & Conquer	Pensamiento recursivo, casos base, call stack	1 semana
08	Algoritmos de Ordenamiento	QuickSort, MergeSort desde cero	1 semana
09	Búsqueda Binaria	Implementación perfecta, variantes	1 semana
16	Dynamic Programming	Memoization, tabulation, problemas clásicos	2 semanas
17	Greedy Algorithms	Cuándo y cómo usar estrategia greedy	1 semana
18	Heaps y Priority Queues	Top K, merge K lists	1 semana

Checkpoint Fase IV: [Simulacro Algoritmos](#)

FASE V: Matemáticas para ML (Mes 5)

Objetivo: Implementar la matemática del ranking sin librerías

#	Módulo	Descripción	Tiempo
10	Álgebra Lineal sin NumPy	Vectores, matrices, operaciones desde cero	2 semanas
11	TF-IDF y Similitud de Coseno	Vectorización de texto, ranking por relevancia	2 semanas

Checkpoint Fase V: [Simulacro Matemáticas](#)

FASE VI: Integración (Mes 6)

Objetivo: Ensamblar todo en un proyecto defendible

#	Módulo	Descripción	Tiempo
12	Proyecto Integrador	Motor de búsqueda completo + análisis Big O	4 semanas

Checkpoint Final: [Simulacro Entrevista Completo](#)



Material Complementario

Práctica y Evaluación

Documento	Descripción
EJERCICIOS.md	Ejercicios prácticos por módulo (3-5 por módulo)
EJERCICIOS_SOLUCIONES.md	Soluciones detalladas con explicación
RUBRICA_EVALUACION.md	Criterios de evaluación (100 puntos)

Referencia

Documento	Descripción
GLOSARIO.md	80+ definiciones técnicas A-Z con analogías
CHECKLIST.md	Verificación final del proyecto

Documento	Descripción
DECISIONES_TECH.md	Por qué Python puro y cada decisión de diseño
REFERENCIAS_CRUZADAS.md	Mapa de navegación entre documentos
EVALUACION_GUIA.md	Autoevaluación de completitud (99/100)

Planificación

Documento	Descripción
SYLLABUS.md	Programa detallado con objetivos y entregables
PLAN_ESTUDIOS.md	Cronograma día a día (6 meses)

Preparación para Entrevistas/Pathway

Documento	Descripción
SIMULACRO_ENTREVISTA.md	50+ preguntas tipo Pathway con respuestas
RECURSOS.md	Cursos, libros, videos recomendados



Proyecto de Referencia

```

archimedes-indexer/
├── src/
│   ├── __init__.py
│   ├── document.py          # Clase Document
│   ├── corpus.py            # Clase Corpus (colección)
│   ├── tokenizer.py         # Tokenización manual
│   ├── inverted_index.py    # Índice invertido (HashMap)
│   ├── sorting.py           # QuickSort, MergeSort
│   ├── searching.py         # Binary Search
│   ├── vectorizer.py        # TF-IDF desde cero
│   ├── similarity.py        # Similitud de coseno
│   └── search_engine.py     # Motor completo
├── tests/
│   ├── test_document.py
│   ├── test_tokenizer.py
│   ├── test_sorting.py
│   ├── test_similarity.py
│   └── test_engine.py
├── docs/
│   └── COMPLEXITY_ANALYSIS.md # Análisis Big O
├── data/
│   └── sample_corpus/        # Documentos de prueba
├── README.md
└── pyproject.toml

```

Mapeo Módulos → Código

Módulos	Archivos del Proyecto
01-02	<code>document.py</code> , <code>corpus.py</code>
04-05	<code>tokenizer.py</code>
06	<code>inverted_index.py</code>
07-08	<code>sorting.py</code>
09	<code>searching.py</code>










Módulos	Archivos del Proyecto
10-11	<code>vectorizer.py</code> , <code>similarity.py</code>
12	<code>search_engine.py</code> , <code>COMPLEXITY_ANALYSIS.md</code>

Tiempo Total Estimado

Componente	Horas
Lectura de módulos	~80 horas
Ejercicios prácticos	~120 horas
Implementación del proyecto	~200 horas
Análisis y documentación	~40 horas
Preparación entrevistas	~40 horas
TOTAL	~480 horas

Con 6 horas/día (L-S) = 36 horas/semana → **~14 semanas efectivas** (margen incluido en 6 meses)

Convenciones de la Guía

Icono	Significado
	Tip o consejo práctico
	Advertencia importante
	Buena práctica
	Anti-patrón a evitar
	Ejercicio práctico
	Objetivo del módulo/sección
	Análisis de complejidad
	Concepto clave para memorizar
	Referencia cruzada

Rutas de Aprendizaje

Ruta Completa (Recomendada)

01 → 02 → 03 → 04 → 05 → 06 → 13 → 14 → 15 → 07 → 08 → 09 → 16 → 17 → 18 → 10 → 11 → 12

Ruta Acelerada (Ya sabes OOP)

03 → 04 → 05 → 06 → 13 → 14 → 15 → 07 → 08 → 09 → 16 → 17 → 18 → 10 → 11 → 12

Ruta Solo Pathway (Foco en DSA) ★

04 → 05 → 13 → 14 → 15 → 07 → 08 → 09 → 16 → 17 → 18 → SIMULACRO_ENTREVISTA



Siguiente Paso

[Comenzar con Módulo 01: Python Profesional →](#)

"Give me a lever long enough and a fulcrum on which to place it, and I shall move the world." — Archimedes

Este proyecto es tu palanca. Los fundamentos son tu fulcro.