Centro Universitário de Excelência
Sistemas de Informação

# Sistema de Gerenciamento Pedidos

**Autor:** Jamile Martins Coutrim

ItabunA, 2025

**1. Introdução:**
Sistema Python para gerenciar cardápio e pedidos usando árvore AVL como indexador para buscas rápidas em dados persistidos JSON.

**2. Funcionalidades Principais:**
Cadastrar itens, mostrar cardápio, realizar pedidos, gerar relatórios ordenados e indexar dados com árvore AVL balanceada.

**3. Processamentos de Pedidos:**
Solicita cliente, exibe cardápio, adiciona itens por ID, finaliza com sair, salva pedido sequencial em JSON arquivo.

**4. Estrutura de Dados:**
JSON armazena itens e pedidos; AVL indexa por ID com nós contendo chave, valor, altura e ponteiros filhos.

**5. Tecnologia Utilizadas e Referência:**
Python, JSON, VS Code, AVL Tree, Insertion Sort com complexidade O(log n) busca e $O(n^2)$ ordenação pior caso.

# Introdução — Indexador AVL no Sistema de Pedidos

- **Contexto:** o sistema gerencia cardápio e pedidos armazenados em JSON; operações frequentes precisam localizar pedidos/itens por ID.
- **Problema:** buscas lineares em listas tornam-se lentas à medida que o volume cresce.
- **Objetivo:** usar um indexador em memória para consultas e inserções rápidas e ordenadas.
- **Solução adotada:** árvore AVL — estrutura binária auto-balanceada que garante complexidade O(log n) em buscas e inserções.
- **Resultado esperado:** respostas mais rápidas, listagens ordenadas sem reordenação constante e melhor escalabilidade do sistema.

# Funcionalidades Principais

**Indexador AVL**
- Classe Node e AVLTree: inserção balanceada (rotações L/R) para manter árvore AVL.
- Função carregar_em_avl: carrega itens e pedidos em árvores AVL para indexação em memória.

**Persistência JSON**
- Arquivo: dados.json
- carregar_dados() e salvar_dados(): ler/gravar itens e pedidos.

**Gerenciamento do cardápio**
- registrar_item(dados): cadastrar novo item (id, nome, preço).
- mostrar_cardapio(dados): listar itens cadastrados.

**Pedidos**
- realizar_pedido(dados): criar pedido, associar cliente e itens (salva em dados.json).
- relatorio_pedidos(dados): listar pedidos ordenados por id (usa insertion_sort).

**Utilitários**
- insertion_sort(lista, key): ordena listas por campo (usado no relatório).
- menu(): interface de console que orquestra todas as opções.

# Processamento de Pedidos

**Solicita nome do cliente** — identifica quem está fazendo o pedido
**Exibe cardápio** — mostra todos os itens disponíveis com ID, nome e preço
**Adiciona itens** — usuário digita o ID do item desejado; o sistema procura linearmente na lista e adiciona ao pedido
**Finaliza pedido** — ao digitar "sair", se houver itens o pedido é salvo em JSON com ID sequencial
**Gera ID automático** — usa str(len(dados["pedidos"]) + 1) para criar ID único

```
pedido = {
    "id": str(len(dados["pedidos"]) + 1),
    "cliente": nome_cliente,
    "itens": []
}

for item in dados["itens"]:
    if item["id"] == escolha:
        pedido["itens"].append(item)
        break

dados["pedidos"].append(pedido)
salvar_dados(dados)
```

# Estrutura de Dados

**Arquivo dados.json (top-level)**

- objetos: {"itens": [...], "pedidos": [...]}

**Itens (cada elemento em dados["itens"])**

- id: string (ex: "1" ou "a1")
- nome: string
- preco: número (float)
-

**Pedidos (cada elemento em dados["pedidos"])**

- id: string (gerado como str(len(pedidos)+1))
- cliente: string
- itens: lista de objetos de item (cópia do dict do cardápio no momento do pedido)

**Indexador AVL**

- Node.key: id (string)
- Node.value: referência ao dict do item ou do pedido
- Árvore usada apenas como índice em memória (busca por id em O(log n) se implementado)

# Tecnologia Utilizada e Referência

Tecnologias utilizadas

- Linguagem: Python (3.x) — script console.
- Persistência: JSON (módulo json da stdlib).
- Editor/Execução: VS Code / terminal.
- Estruturas de dados implementadas: Árvore AVL (indexador), insertion sort, listas como filas FIFO.
- Arquivos: dados.json (itens + pedidos) no filesystem.

Referências

Aulas Gravadas
Livro: entendendo algoristimos