Aula 1

Estrutura de dados Lineares

Prof. Wesley Gonzaga Alves

ATAL

UNIFACISA
CENTRO UNIVERSITÁRIO

SISTEMAS DE INFORMAÇÃO

±80 anos atrás...

Os primeiros programas não eram complexos.

O acesso era **sequencial**, ou seja, os dados precisavam ser lidos em uma **ordem específica**, o que tornava a recuperação de dados lenta, especialmente em posições distantes da sequência.

Com o aumento da complexidade...

Ficou claro que a forma como os dados eram organizados impactava diretamente a eficiência dos algoritmos.

Então surgiram as primeiras **estruturas de dados**:

Linear

Listas, pilhas, filas e deques, organizam os elementos em sequência, onde cada elemento tem **um único sucessor e um único antecessor**, facilitando operações como inserção e remoção.

Não lineares

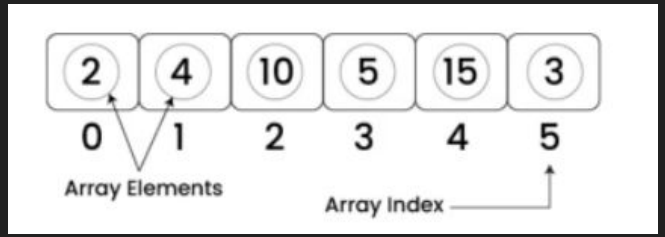
Organizam os dados de forma mais complexa, **permitindo múltiplos caminhos entre os elementos**. Exemplos incluem **árvores e grafos.**

Para cada estrutura de dados…

Temos duas formas diferentes de realizarmos a implementação

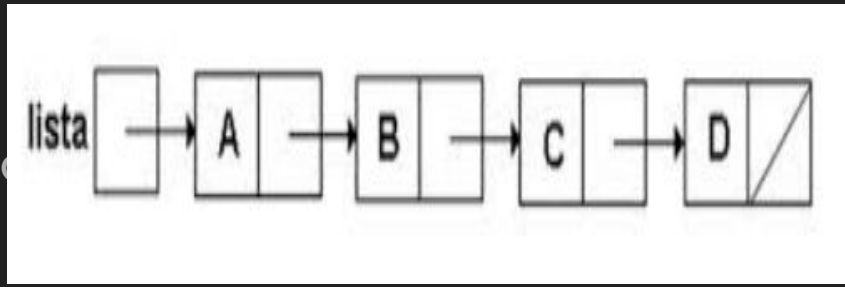
1. Sequencial

   bloco contínuo de memória para armazenar elementos



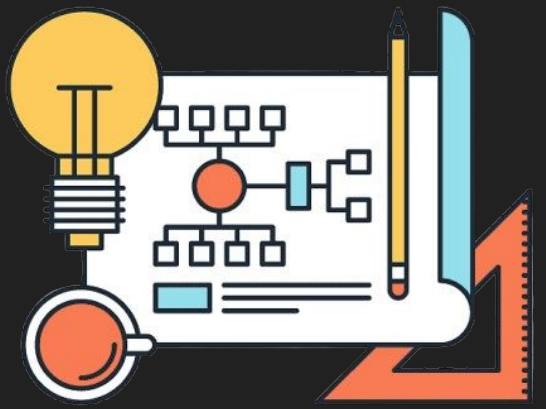
2. Encadeada

   Cada elemento possui referênci
   próximo nó na sequência



Ponto de partida

1. Estrutura de Dados Lineares
    - Listas
    - Filas
    - Pilhas
    - Deques
2. Implementações
    - Sequenciais
    - Encadeadas



LISTAS

Características:

- Cada elemento tem uma posição específica;
- Armazenamento sequencial;
- Dinâmica (Sem tamanho) ou estática (Tamanho fixo);
- Pode ser desordenada ou ordenada.

LISTAS

Principais operações:

Inserção e remoção permite adicionar ou excluir elementos no início, fim ou em posições;

Localização de itens para acesso por índice ou buscar se um determinado elemento existe;

Exemplos:

Listas de compras, Registro de alunos, Sumário de um livro, Agenda de contatos, Logs de atividades...

LISTAS

Sequencial

Pontos positivos	Pontos negativos
Acesso direto e rápido aos elementos *O(1)*	Tamanho fixo, requer realocação para aumentar o tamanho
Eficiência na busca por índice	Inserção e remoção custosas em posições intermediárias *O(n)*

LISTAS

Encadeada

Pontos positivos	Pontos negativos
Tamanho dinâmico, pode crescer conforme necessário	Acesso sequencial, lento para buscar por índice *O(n)*
Inserções e remoções rápidas no início e no meio *O(1)*	Requer memória extra para armazenar os ponteiros

FILAS

Características:

- FIFO (First In, First Out);
- Primeiro elemento a entrar na fila é o primeiro a sair;
- Fila comum (Queue) e Fila de prioridade (Priority Queue);

Principais operações:

- Enfileirar *(**enqueue**)* - Adicionar um elemento ao final da fila;
- Desenfileirar *(**dequeue**)* - Remover o elemento do início da fila;
- Visualizar o primeiro ou último elemento da fila sem remover;
- Verificar se está vazia;

FILAS

Exemplos:

- Enfileirar mensagens para processamento.
- Gerenciar eventos de interface gráfica na ordem de ocorrência.
- Streaming de Dados
- Testes automatizados

FILAS

Sequencial

Pontos positivos	Pontos negativos
Acesso rápido ao fim da fila para inserção e início para remoção O(1)	Remoção de elementos pode ser ineficiente, com deslocamento dos elementos subsequentes O(n)
Boa para filas de tamanho fixo	Tamanho fixo, realocação necessária para aumentar a capacidade

FILAS

Encadeada

Pontos positivos	Pontos negativos
Inserção e remoção eficientes em ambos os extremos *O(1)*	Requer memória extra para os ponteiros
Tamanho dinâmico	Acesso sequencial, sem acesso direto aos elementos

PILHAS

Características:

- LIFO (Last In, First Out);
- Último elemento adicionado é o primeiro a ser removido;

Principais operações:

- Empilhar *(push)* - Adiciona no topo da lista
- Desempilhar *(pop)* - Remove do topo da lista
- Visualizar *(peek ou top)* o primeiro elemento da pilha;
- Verificar se está vazia;

PILHAS

Exemplos:

- Controlar histórico de navegação;
- Interpretação de Código *(Debug)*;
- Gerenciamento de memória alocação e liberação de memória;

PILHAS

Sequencial

Pontos positivos	Pontos negativos
Inserção e remoção rápidas no topo *O(1)*	Tamanho fixo, necessita realocação para aumentar a capacidade
Acesso rápido ao topo da pilha	Não permite inserção e remoção no meio

PILHAS

Encadeada

Pontos positivos	Pontos negativos
Inserção e remoção eficientes no topo *O(1)*	Requer memória extra para armazenar ponteiros
Tamanho dinâmico	

DEQUES

Características:

- Double-ended Queues;
- Inserções e remoções de elementos nas extremidades da lista;
- Maior flexibilidade em relação a filas e pilhas;

DEQUES

Principais operações:

- Inserir no início *(addFirst)*;
- Inserir no final *(addLast)*;
- Remoção do início (*removeFirst*);
- Remoção do fim (*removeLast*);
- Visualizar o elemento no início *(peekFirst);*
- Visualizar o elemento no fim *(peekLast).*

DEQUES

Exemplos:

- Capturar teclas de ambos os extremos;
- Agendamento de processos;
- Gerenciar feed de notícias em um aplicativo (Sliding Window);

DEQUES

Sequencial

Pontos positivos	Pontos negativos
Inserção e remoção eficientes nas extremidades *O(1)*	Inserção e remoção no meio podem ser ineficientes *O(n)*
	Tamanho fixo, realocação necessária

DEQUES

Encadeada

Pontos positivos	Pontos negativos
Inserção e remoção rápidas em ambos os extremos *O(1)*	Requer memória extra para os ponteiros
Tamanho dinâmico	Acesso sequencial, sem acesso direto

Quando, como e por que usar?

Próxima aula

Collections Framework