Matéria: Tecnologia da Informação

Assunto: Algoritmos, Estruturas de Dados e Banco de Dados

Resumo Teórico do Assunto

Excelente! Para dominar as questões sobre Algoritmos, Estruturas de Dados e Banco de Dados, é crucial entender os fundamentos por trás de cada conceito. Vamos desmistificar esses tópicos:

Algoritmos, Estruturas de Dados e Banco de Dados: Fundamentos Essenciais

A Tecnologia da Informação (TI) é construída sobre a capacidade de processar, armazenar e recuperar dados de forma eficiente. Isso é alcançado através de **algoritmos** que operam sobre **estruturas de dados** e, em muitos casos, interagem com **bancos de dados**.

---

1. Algoritmos: A Receita para Resolver Problemas

Um **algoritmo** é um conjunto finito e bem definido de instruções passo a passo para resolver um problema ou realizar uma tarefa. A eficiência de um algoritmo é medida por seu **consumo de tempo** e **espaço** (memória).

 # 1.1. Análise de Complexidade (Notação Big O - O)

A **Notação Big O** é usada para descrever o desempenho ou a complexidade de um algoritmo. Ela indica como o tempo de execução ou o espaço de memória cresce à medida que o tamanho da entrada (n) aumenta.

• **O(1) - Tempo Constante:** O tempo de execução é o mesmo, independentemente do tamanho da entrada. Ex: Acessar um elemento em um array pelo índice.
• **O(log n) - Tempo Logarítmico:** O tempo de execução cresce lentamente com o tamanho da entrada. Comum em algoritmos que dividem o problema pela metade a cada passo. Ex: Busca binária.
• **O(n) - Tempo Linear:** O tempo de execução cresce proporcionalmente ao tamanho da entrada. Ex: Percorrer uma lista para encontrar um elemento.
• **O(n log n) - Tempo Linear-Logarítmico:** Comum em algoritmos de ordenação eficientes. Ex: Merge Sort, Quick Sort.
• **O(n^2) - Tempo Quadrático:** O tempo de execução cresce com o quadrado do tamanho da entrada. Comum em algoritmos com laços aninhados. Ex: Bubble Sort, Insertion Sort (pior

caso).

É importante considerar o **melhor caso**, **pior caso** e **caso médio** de um algoritmo, pois o desempenho pode variar dependendo da organização dos dados de entrada.

 # 1.2. Algoritmos de Busca

São algoritmos usados para encontrar um item específico dentro de uma coleção de dados.

• Busca Sequencial (Linear Search):
* **Como funciona:** Percorre a coleção de dados item por item, do início ao fim, comparando cada item com o valor procurado.
* **Desempenho:**
* **Dados Desordenados:**
* **Melhor Caso:** 1 comparação (o item procurado é o primeiro).
* **Pior Caso:** `n` comparações (o item é o último ou não está na lista).
* **Caso Médio:** `n/2` comparações.
* **Dados Ordenados (Busca Sequencial Otimizada):**
* Funciona da mesma forma, mas pode parar a busca mais cedo se o item atual for maior que o item procurado (pois o item procurado não estará mais à frente na lista ordenada).
* **Melhor Caso:** 1 comparação (o item é o primeiro, ou o item procurado é menor que o primeiro elemento da lista, indicando que não está presente).
* **Pior Caso:** `n` comparações (o item é o último, ou não está na lista e é maior que todos os elementos).
* **Conhecimento Essencial para Q63:** O "menor número de comparações" para *conhecer o resultado* da busca ocorre quando o item é encontrado logo no início (1 comparação) ou quando a busca pode ser interrompida rapidamente porque o item não pode mais estar na lista (ex: lista ordenada e o valor atual já é maior que o procurado).

 # 1.3. Algoritmos de Ordenação

São algoritmos usados para organizar uma coleção de dados em uma ordem específica (crescente ou decrescente).

• Insertion Sort (Ordenação por Inserção):
* **Como funciona:** Constrói a lista ordenada um item por vez. Ele itera sobre os elementos da entrada e insere cada elemento em sua posição correta na parte já ordenada da lista. É como organizar um baralho de cartas na mão.
* **Desempenho:**
* **Melhor Caso: O(n)** - Ocorre quando a lista já está **quase ou totalmente ordenada**. Cada elemento precisa de apenas uma comparação para confirmar sua posição.
* **Pior Caso: O(n²)** - Ocorre quando a lista está ordenada em ordem inversa. Cada elemento precisa ser comparado e movido através de quase toda a parte já ordenada.
* **Caso Médio: O(n²)**.
* **Conhecimento Essencial para Q64:** O foco é no **melhor caso** do Insertion Sort, que é

O(n).

---

2. Estruturas de Dados: Organizando a Informação

Estruturas de Dados são formas específicas de organizar e armazenar dados em um computador para que possam ser acessados e modificados de forma eficiente. Exemplos incluem arrays, listas encadeadas, árvores, pilhas, filas e **grafos**.

---

3. Banco de Dados: Persistência e Gerenciamento de Dados

Um **Banco de Dados** é uma coleção organizada de informações (dados) que são armazenadas e acessadas eletronicamente.

 # 3.1. Bancos de Dados NoSQL

NoSQL (Not Only SQL) é uma categoria de sistemas de gerenciamento de banco de dados que não seguem o modelo relacional tradicional (SQL). Eles são projetados para lidar com grandes volumes de dados não estruturados ou semi-estruturados, oferecer alta escalabilidade e flexibilidade de esquema.

• **Tipos Comuns de NoSQL:**
* **Key-Value Stores:** Armazenam dados como pares chave-valor simples.
* **Document-Oriented Databases:** Armazenam dados em documentos (ex: JSON, BSON, XML), que podem ter estruturas variadas.
* **Column-Family Databases:** Armazenam dados em colunas, otimizados para grandes volumes de dados distribuídos.
* **Graph-Oriented Databases (Bancos de Dados Orientados a Grafos):**
* **Como funcionam:** Armazenam dados em uma estrutura de **grafo**, que consiste em **nós (ou vértices)** e **arestas (ou relacionamentos)**.
* **Nós (Objetos - O):** Representam entidades (pessoas, lugares, eventos, produtos). Podem ter **propriedades** (atributos).
* **Arestas (Relacionamentos - R):** Representam as conexões ou interações entre os nós. Também podem ter **propriedades** (ex: data de um relacionamento).
* **Vantagens:** Extremamente eficientes para modelar e consultar dados altamente interconectados, onde as relações são tão importantes quanto os próprios dados.
* **Casos de Uso Típicos:** Redes sociais (amizades, seguidores), sistemas de recomendação, detecção de fraudes, gerenciamento de redes, grafos de conhecimento.
* **Conhecimento Essencial para Q65:** A descrição "dados são organizados em vértices ou objetos (O) e em relacionamentos, que são relações (R) ou arestas" e o exemplo fornecido (`O:Usuario`, `O:Escola`, `R:Estudaem`, `R:Amigode`) são a definição exata de um **Banco de Dados Orientado a Grafos**.

Dominando esses conceitos, você estará bem preparado para abordar questões que envolvem a eficiência de algoritmos, a escolha da estrutura de dados correta e o tipo de banco de dados mais adequado para diferentes cenários.

Questões de Provas Anteriores

Fonte: escriturario_agente_de_tecnologia (1).pdf, Página: 24

pcimarkpci MjgwNDowMTRkOjE0YTU6OTI1ODozOGQ2OjNhMGM6NTM0MzplZmI1:U3VuLCAyNyBKdWwgMjAyNSAyMzo0Nzo0MCAtMDMwMA==

BANCO DO BRASIL
AGENTE DE TECNOLOGIA - Microrregião 16 DF-TI GABARITO 1
63
Em uma agência bancária, as filas de atendimento são ordenadas da esquerda para a direita, e o gerente dessa agência percebeu a presença equivocada de um idoso, com a senha 52, na fila de atendimento não preferencial. Visando a sanar o equívoco, o gerente resolveu que, na primeira oportunidade, faria uma busca no sistema para saber se a senha 52 ainda estava ativa, indicando a presença do idoso na fila de atendimento não preferencial. Em caso de resposta positiva, procuraria o cliente para trocar sua senha por outra de atendimento preferencial; se não, apenas registraria o fato para posterior discussão no grupo de qualidade de atendimento.
Considerando o uso de um algoritmo de busca sequencial otimizado, partindo da esquerda para a direita, e as sequências hipotéticas das senhas da fila de atendimento não preferencial e suas regras de ordenação, segundo as quais quem está à esquerda é atendido antes de quem está à direita, o menor número de comparações para o gerente conhecer o resultado de sua busca ocorre em
Regras de ordenação Sequência das senhas na fila de atendimento não preferencial

(A) Sequência ordenada crescentemente 23; 45; 81; 97; 112; 138; 154
(B) Sequência ordenada crescentemente 13; 25; 37; 44; 52; 78; 83; 91

(C) Sequência ordenada crescentemente 17; 28; 32; 49; 67; 85; 94; 103
(D) Sequência desordenada 27; 95; 148; 117; 33; 59; 52
(E) Sequência desordenada 32; 48; 12; 55; 93; 27; 66

64

Dentre os problemas identificados pela gerência de um banco comercial, está a localização das contas dos seus titulares
nas listagens e nos relatórios impressos em diferentes situações. Um especialista de TI sugeriu ordenar as contas por meio
dos CPF dos seus n titulares antes das impressões.
Dentre alguns algoritmos pré-selecionados para essa ordenação, o especialista escolheu o algoritmo de ordenação por
inserção, no qual o consumo de tempo é, no melhor caso, proporcional a
(A) n log n
(B) log n
(C) n
2
(D) n
(E) 1

65

Um banco comercial deseja obter um tipo de banco de dados NoSQL que trate os dados extraídos de redes sociais, de
modo a formar uma coleção (collection) interconectada. Nessa coleção (collection), os dados são organizados em vértices
ou objetos (O) e em relacionamentos, que são relações (R) ou arestas.
Nesse modelo de banco de dados NoSQL, os dados seriam apresentados da seguinte forma:
O:Usuario{u1:Joao, u2:Jose, u3:Maria, u4:Claudio}
O:Escola{e1:UFRJ, e2:URGS, e3:IFB}
R:Estudaem{re1=u1:e2;re2=u2:e2;re3=u3:e1;re4=u4:e3}
R:Amigode{ra1=u1:u2;ra2=u1:u3;ra3=u2:u3}
O banco de dados NoSQL que representa essa situação deve ter uma estrutura do tipo
(A) Distribuided Hashing
(B) Consistent Hashing
(C) Document Oriented
(D) Graph Oriented
(E) Vector Clock