UNIVERSIDADE DO ESTADO DO RIO DE JANEIRO
INSTITUTO POLITÉCNICO
Engenharia de Computação — Disciplina: Matemática Discreta II
Trabalho 1
Data da apresentação: 05/05/2023 — Data da entrega: 15/06/2023

Contexto

Imagine que um banco precisa armazenar informações de seus clientes, como nome, endereço, número de telefone, saldo da conta etc. Em vez de armazenar essas informações em uma lista ou matriz, que seria ineficiente em termos de tempo de busca de uma chave e espaço de armazenamento, o banco pode usar tabelas *hash* (ou tabelas de dispersão) para armazenar estas informações.

Cada cliente pode ser representado como uma chave de *hash*, que é gerada a partir de seu número de conta. A tabela *hash* pode então ser usada para mapear cada chave de *hash* para uma posição na tabela. No entanto, pode haver colisões, ou seja, duas chaves de *hash* diferentes podem ser mapeadas para a mesma posição na tabela (sinônimos). Para resolver as colisões, o banco pode usar técnicas de tratamento de colisões.

Problema

Suponha que temos uma tabela de dispersão com 9 posições, numeradas de 1 a 9, e queremos armazenar informações de clientes. Vamos usar o número de conta de cada cliente como chave de *hash* para mapeá-lo na tabela de dispersão. As chaves de *hash* são geradas a partir dos últimos três dígitos do número de conta. Cada conta é sempre formada por 5 dígitos, representados por números inteiros que variam entre 10000 e 99999.

Por exemplo, se o número da conta de um cliente é 34567, a chave de *hash* para este cliente é o resto da divisão de 567 por 9, que é 0. Então, armazenamos as informações do cliente na posição 1 da tabela de dispersão. Se o número da conta fosse, entretatno, 34568, o resto da divisão de 568 por 9 é 1 e, portanto, este cliente teria suas informações armazenadas na posição 2.

Agora, para um cliente cujo número de conta é 23459, a chave de *hash* também é 1, já que os últimos três dígitos do número de conta são 459 e o resto da divisão deste número por 9 é igual a 0. No entanto, a posição 1 já está ocupada pelo primeiro cliente, então temos uma colisão. Para resolver este problema, podemos usar diferentes técnicas de tratamento de colisões.

Proposta

- Gere um conjunto de, pelo menos, 1000 números de conta aleatórios, com distribuição uniforme no intervalo entre 10000 e 99999;
- Implemente uma tabela *hash* e armazene as contas dos clientes de acordo com as regras definidas anteriormente;
- Use dois métodos de tratamento de colisão a sua escolha, visando evitar que dados sejam perdidos quando duas chaves arbitrárias forem sinônimo.

Análises

- Qual é a diferença entre os métodos de tratamento de colisão adotados? Eles produzem os mesmos resultados?
- Qual é o custo de se encontrar uma chave específica com cada estratégia escolhida?
- Se a chave de *hash* fosse definida considerando-se todos os dígitos do número da conta, ao invés de apenas os útlimos três, a busca por uma chave seria mais rápida?

Regras

- O trabalho será em dupla e representa 25% da nota total da disciplina;
- Deve ser entregue:
  - Relatório contendo (1) fundamentação teórica sobre tabelas *hash* e as estratégias de tratamento de colisão escolhidas; (2) explicação do código-fonte implementado e; (3) resultados obtidos e análises relacionadas.
  - Código executável, elaborado na linguagem da sua escolha, que garanta a reprodutibilidade dos resultados alcançados.
- O relatório deve ser encaminhado (preferencialmente) na página da disciplina no Moodle em formato *pdf*; o código-fonte pode ser encaminhado da mesma forma (ambos em arquivo comprimido) ou depositado em um repositório (preferencialmente) e reportado no relatório.