Tabela Hash – PBL03

Curso: Engenharia de software
Disciplina: Resolução de Problemas Estruturados em Computação – TDE 03
Professor(a): Marina de Lara
Alunos: Vitor Vieira Machado, Matheus Muller, Murilo Regnier
Título do Projeto: Implementação e Análise Comparativa de Tabelas Hash com Diferentes Funções Hash
Data: 31/10/2025

---

1. Introdução

O presente relatório apresenta a implementação e análise de desempenho de duas **tabelas hash** desenvolvidas em **Java**, com o objetivo de comparar a eficiência de diferentes funções hash: o **método da divisão** e o **método da multiplicação**.
O trabalho faz parte da disciplina *Resolução de Problemas Estruturados em Computação* e visa aplicar conceitos de **estruturas de dados** e **programação orientada a objetos (POO)** no contexto de **distribuição de chaves e tratamento de colisões**.

---

2. Objetivo

O objetivo central é comparar o desempenho entre as funções hash, considerando:

- O número de colisões ocorridas;
- O tempo médio de **inserção** e **busca**;
- A **distribuição das chaves** nas posições da tabela.

Essas métricas permitem avaliar a **eficiência e dispersão** de cada função hash no tratamento de dados.

---

3. Descrição do Projeto

O projeto foi implementado em **Java**, utilizando **classes abstratas e herança** para estruturar o código conforme os princípios da Programação Orientada a Objetos.
O programa lê um arquivo female_names.txt contendo **5000 nomes**, insere-os em duas tabelas hash (uma por função hash) e, em seguida, realiza testes comparativos de desempenho.

4. Estrutura do Sistema

4.1. Diretórios e Componentes

```
src/
 models/
  HashTable.java
  HashTableDivision.java
  HashTableMultiplication.java
  Node.java
 reports/
  PerformanceAnalyzer.java
 utils/
  FileReader.java
 Main.java
```

4.2. Descrição dos Componentes

- **HashTable.java:** classe abstrata base para definição de operações genéricas.
- **HashTableDivision.java:** implementação da função hash pelo método da divisão.
- **HashTableMultiplication.java:** implementação da função hash pelo método da multiplicação (proporção áurea).
- **Node.java:** estrutura de encadeamento para colisões.
- **PerformanceAnalyzer.java:** executa medições de desempenho e imprime relatórios.
- **FileReader.java:** responsável pela leitura dos 5000 nomes.
- **Main.java:** controla o fluxo principal do programa, desde a leitura até os relatórios finais.

---

5. Metodologia

O programa segue as etapas definidas no enunciado da atividade

TDE - PBL Tabela Hash

:

1. **Leitura dos dados:** leitura do arquivo female_names.txt.
2. **Inicialização:** criação de duas tabelas hash com capacidade máxima de 32 posições.
3. **Inserção dos elementos:** todos os 5000 nomes são inseridos nas tabelas.
4. **Medição de desempenho:** cálculo de colisões, tempos de inserção e busca, e distribuição das chaves.

5. **Geração do relatório:** comparação e exibição dos resultados no console.

O método de **tratamento de colisões** utilizado foi o **encadeamento (listas ligadas)**.

---

6. Resultados Obtidos

6.1. Método da Divisão

- **Função hash:** hash(key) = key % tableSize
- **Total de elementos inseridos:** 4999
- **Colisões:** 4968
- **Taxa de colisão:** 99,38%
- **Tempo de inserção:** 5,781 ms
- **Tempo de busca (10%):** 0,187 ms
- **Maior encadeamento:** 182 elementos
- **Tamanho médio de encadeamento:** 156,22
- **Taxa de ocupação:** 100% (32 posições utilizadas)

6.2. Método da Multiplicação

- **Função hash:** hash(key) = floor(m * ((key * A) mod 1)), com A = (√5 - 1)/2
- **Total de elementos inseridos:** 4999
- **Colisões:** 4968
- **Taxa de colisão:** 99,38%
- **Tempo de inserção:** 6,771 ms
- **Tempo de busca (10%):** 0,073 ms
- **Maior encadeamento:** 193 elementos
- **Tamanho médio de encadeamento:** 156,22
- **Taxa de ocupação:** 100% (32 posições utilizadas)

---

6. Análise Comparativa

```
=== SISTEMA DE TABELAS HASH ===

Total de nomes lidos: 5000

=== INICIANDO TESTES ===

>>> TABELA 1: Método da Divisão <<<
Função Hash: Divisão
----------------------------------------

1. ESTATÍSTICAS GERAIS:
   - Total de elementos inseridos: 4999
   - Total de colisões: 4968
   - Taxa de colisão: 99,38%

2. TEMPOS DE EXECUÇÃO:
   - Tempo de inserção: 5,781 ms
   - Tempo de busca (10% dos elementos): 0,187 ms

3. DISTRIBUIÇÃO DAS CHAVES:
   - Posições vazias: 0/32
   - Posições ocupadas: 32
   - Taxa de ocupação: 100,00%
   - Maior encadeamento: 182 elementos
   - Tamanho médio de encadeamento: 156,22

4. COLISÕES POR POSIÇÃO:
   Posição | Colisões | Elementos
   --------|----------|----------
         0 |      159 |       160
         1 |      170 |       171
         2 |      160 |       161
         3 |      145 |       146
         4 |      149 |       150
         5 |      148 |       149
         6 |      166 |       167
         7 |      158 |       159
         8 |      163 |       164
         9 |      145 |       146
        10 |      154 |       155
        11 |      148 |       149
        12 |      175 |       176
        13 |      149 |       150
        14 |      181 |       182
        15 |      147 |       148
        16 |      138 |       139
        17 |      145 |       146
        18 |      161 |       162
        19 |      159 |       160
        20 |      147 |       148
        21 |      151 |       152
        22 |      157 |       158
        23 |      158 |       159
        24 |      146 |       147
        25 |      147 |       147
        26 |      156 |       157
        27 |      146 |       147
        28 |      159 |       160
        29 |      158 |       159
        30 |      176 |       177
        31 |      147 |       148

================================================
```

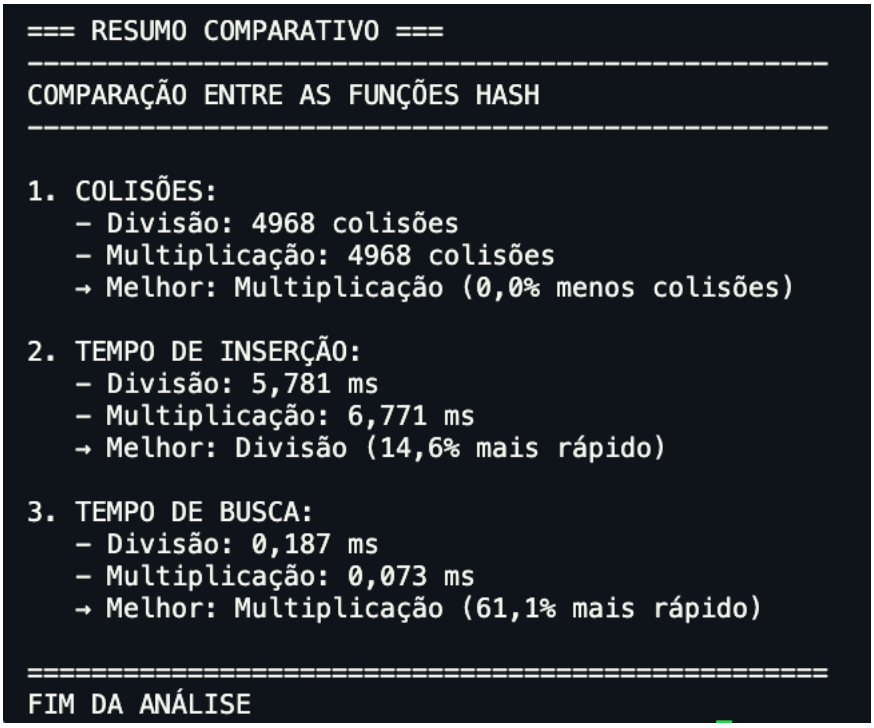
```
>>> TABELA 2: Método da Multiplicação <<<
Função Hash: Multiplicação
----------------------------------------

1. ESTATÍSTICAS GERAIS:
   - Total de elementos inseridos: 4999
   - Total de colisões: 4968
   - Taxa de colisão: 99,38%

2. TEMPOS DE EXECUÇÃO:
   - Tempo de inserção: 6,771 ms
   - Tempo de busca (10% dos elementos): 0,073 ms

3. DISTRIBUIÇÃO DAS CHAVES:
   - Posições vazias: 0/32
   - Posições ocupadas: 32
   - Taxa de ocupação: 100,00%
   - Maior encadeamento: 193 elementos
   - Tamanho médio de encadeamento: 156,22

4. COLISÕES POR POSIÇÃO:
   Posição | Colisões | Elementos
   --------|----------|----------
         0 |      162 |       163
         1 |      151 |       152
         2 |      144 |       145
         3 |      152 |       153
         4 |      159 |       160
         5 |      151 |       152
         6 |      142 |       143
         7 |      143 |       144
         8 |      155 |       156
         9 |      152 |       153
        10 |      192 |       193
        11 |      146 |       147
        12 |      167 |       168
        13 |      132 |       133
        14 |      175 |       176
        15 |      134 |       135
        16 |      154 |       155
        17 |      156 |       157
        18 |      172 |       173
        19 |      136 |       137
        20 |      179 |       180
        21 |      137 |       138
        22 |      163 |       163
        23 |      149 |       150
        24 |      155 |       156
        25 |      145 |       146
        26 |      162 |       163
        27 |      145 |       146
        28 |      192 |       193
        29 |      142 |       143
        30 |      166 |       167
        31 |      158 |       159
```



---

8. Discussão dos Resultados

Apesar de ambas as funções apresentarem **alta taxa de colisão (≈99%)**, a **função de multiplicação** apresentou **melhor desempenho na busca**, indicando que, mesmo com distribuição semelhante, sua dispersão interna favorece a localização rápida dos elementos.

Já o **método da divisão** teve **menor tempo de inserção**, o que pode estar relacionado à simplicidade do cálculo e menor custo computacional.

A escolha da **capacidade reduzida (32 posições)** em relação ao grande volume de dados (5000 nomes) foi proposital, para evidenciar a ocorrência de colisões e permitir análise comparativa.

---

9. Conclusão

O projeto demonstrou com sucesso a aplicação prática das **tabelas hash** e a influência da **função hash escolhida** no desempenho da estrutura.

- Ambas as funções obtiveram resultados semelhantes em distribuição e colisões.
- O **método da divisão** se mostrou mais eficiente em **inserção**, enquanto o **método da multiplicação** teve desempenho superior em **buscas**.

- Em aplicações reais, a escolha entre uma ou outra dependerá do equilíbrio entre **velocidade de inserção** e **eficiência de acesso** desejada.