Resolução de Problemas Estruturados em Computação
RA3- Tabelas Hash **Prof: Andrey Cabral Meira**
Aluno: João Vitor Zambão **BCC turma: 4B**

Link: https://github.com/JVZambao/Ra3_Andrey/tree/main/TabelaHash

1. Descrição de cenário:

1.1. Neste trabalho pediu-se que desenvolvêssemos algoritmos de Tabela Hash, na linguagem Java, o tipo a qual ficou ao critério dos alunos, para primeiramente comprovar o aprendizado do conteúdo e para testar a efetividade dos métodos normalmente utilizados.

1.2. Em meu projeto foram utilizados os algoritmos: Dobramento, Multiplicação e Divisão. Feito no NetBeans.

Método de Divisão:

- **Funcionamento:**
  - O método de divisão envolve calcular o índice da tabela hash usando o resto da divisão da chave pelo tamanho da tabela.
  - A fórmula geral é: **índice = chave % tamanho_da_tabela**.
  - A chave é dividida pelo tamanho da tabela, e o resto dessa divisão é usado como índice.
- **Prós e Contras:**
  - **Prós:** Simplicidade e eficácia se o tamanho da tabela for escolhido corretamente.
  - **Contras:** Pode ter problemas se o tamanho da tabela não for escolhido de maneira apropriada, levando a colisões frequentes.

Dados:

Tamanho da Tabela	Tamanho do Conjunto	Tempo de Inserção Divisão (ms)	Colisões Inserção Divisão
10	20,000	4	19,990
100	20,000	2	19,900
1,000	20,000	2	19,000
10,000	20,000	2	11,372
100,000	20,000	0	1,944
10	100,000	1	99,990
100	100,000	0	99,900
1,000	100,000	2	99,000
10,000	100,000	4	90,000
100,000	100,000	3	36,678
10	500,000	0	499,990
100	500,000	5	499,900
1,000	500,000	14	499,000
10,000	500,000	8	490,000
100,000	500,000	29	400,666
10	1,000,000	11	999,990
100	1,000,000	11	999,900
1,000	1,000,000	23	999,000
10,000	1,000,000	28	990,000
100,000	1,000,000	77	900,006
10	5,000,000	76	4,999,990
100	5,000,000	55	4,999,900
1,000	5,000,000	262	4,999,000
10,000	5,000,000	202	4,990,000
100,000	5,000,000	1151	4,900,000

Método de Multiplicação:

- **Funcionamento:**
  - Este método utiliza operações de multiplicação e fração para calcular o índice da tabela hash.
  - A fórmula geral é: **índice = ⌊tamanho_da_tabela * (chave * constante_A % 1)⌋**, onde **constante_A** é um valor entre 0 e 1.
- **Prós e Contras:**
  - **Prós:** Boa distribuição de chaves se a constante de multiplicação for escolhida adequadamente.
  - **Contras:** Pode ser mais complexo e requerer cuidado na escolha da constante de multiplicação.

Dados:

Tamanho da Tabela	Tamanho do Conjunto	Tempo de Inserção Multiplicação (ms)	Colisões Inserção Multiplicação
10	20,000	4	19,990
100	20,000	2	19,900
1,000	20,000	1	19,094
10,000	20,000	3	19,094
100,000	20,000	1	19,094
10	100,000	0	99,990
100	100,000	1	99,900
1,000	100,000	4	99,077
10,000	100,000	2	99,077
100,000	100,000	0	99,077
10	500,000	19	499,990
100	500,000	98	499,900
1,000	500,000	11	499,076
10,000	500,000	13	499,076
100,000	500,000	16	499,076
10	1,000,000	91	999,990
100	1,000,000	13	999,900
1,000	1,000,000	26	999,076
10,000	1,000,000	22	999,076
100,000	1,000,000	17	999,076
10	5,000,000	116	4,999,990
100	5,000,000	196	4,999,900
1,000	5,000,000	83	4,999,076
10,000	5,000,000	129	4,999,076
100,000	5,000,000	336	4,999,076

Método de Dobramento (ou Folding):

- **Funcionamento:**
  - Neste método, a chave é dividida em partes menores e essas partes são somadas para obter o índice.
  - Pode envolver somar partes consecutivas, inverter partes, etc.
  - A fórmula geral depende da estratégia escolhida.
- **Prós e Contras:**
  - **Prós:** Simples e fácil de implementar.
  - **Contras:** Pode resultar em colisões se a técnica de dobramento não for escolhida cuidadosamente.

Dados:

Tamanho da Tabela	Tamanho do Conjunto	Tempo de Inserção Dobramento (ms)	Colisões Inserção Dobramento
10	20,000	10	19,990
100	20,000	6	19,900
1,000	20,000	5	19,094
10,000	20,000	8	19,094
100,000	20,000	9	19,094
10	100,000	13	99,990
100	100,000	32	99,900
1,000	100,000	9	99,077
10,000	100,000	17	99,077
100,000	100,000	21	36,689
10	500,000	131	499,990
100	500,000	78	499,900
1,000	500,000	161	499,076
10,000	500,000	144	490,000
100,000	500,000	86	400,736
10	1,000,000	198	999,990
100	1,000,000	143	999,900
1,000	1,000,000	150	999,000
10,000	1,000,000	179	990,000
100,000	1,000,000	180	900,005
10	5,000,000	927	4,999,990
100	5,000,000	674	4,999,900
1,000	5,000,000	1051	4,999,000
10,000	5,000,000	1228	4,990,000
100,000	5,000,000	1870	4,900,000

Conclusões acerca dos diferentes métodos:

Semelhanças:

No senário observado, quando o tamanho dos conjuntos é pequeno, o tempo de inserção dos três métodos se tornam negligenciáveis, assim como o número de colisões, os quais apresentam leves alterações entre eles.

Diferenças:

No senário observado, percebesse que respectivamente os mais eficientes em questão de tempo são: Multiplição; Divisão; e Dobramento;

Em quesitos de eficiência em questão do menor número de colisões eles são respectivamente: Divisão; Dobramento; e Multiplicação;

Conclusão Final:

Com base nos dados adquiridos, em uma situação em que o tamanho do conjunto é pequeno, os métodos podem ser alternados sem muita alteração dos resultados. Porém, em casos que o tamanho do conjunto é maior, o método de Multiplicação é o mais efetivo em questão de tempo, e o de Divisão é o mais efetivo na questão de possuir o menor número de colisões.