INSTITUTO FEDERAL CATARINENSE

CAMPUS VIDEIRA

Vitor Carlet

TABELA HASH EM C

Videira/SC - 2023

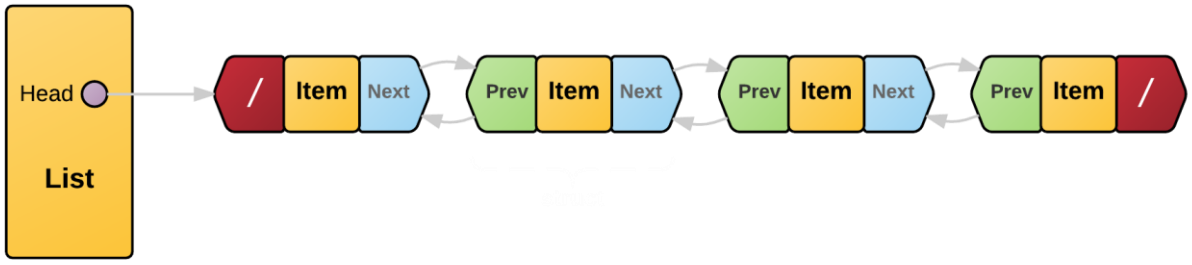
Relatório - Implementação de Tabela Hash com Listas Encadeadas

Introdução:

Este relatório descreve a implementação de uma tabela hash utilizando listas encadeadas em linguagem C. O objetivo é armazenar nomes em diferentes listas, com base em uma função de hash, para permitir uma recuperação eficiente das informações.

Metodologia:

Estruturas de Dados:



Nodo: Representa um elemento da lista encadeada, com ponteiros para o próximo e o anterior, além de um campo para armazenar o nome.

Lista: Mantém ponteiros para o início e o fim da lista, e também registra o tamanho da lista.

Inicialização da Tabela Hash:

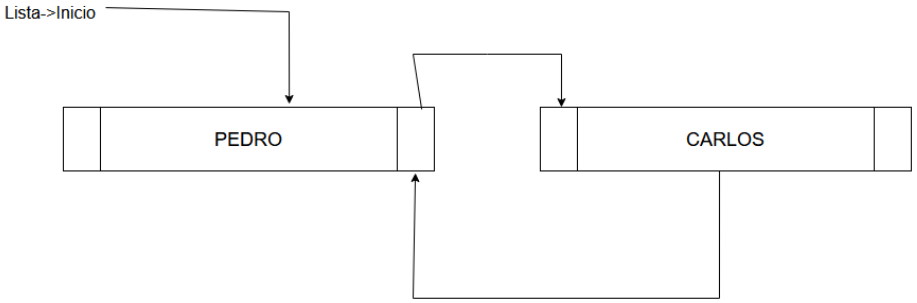
A função "inicializarTabela()" percorre todos os elementos da tabela e inicializa cada lista.

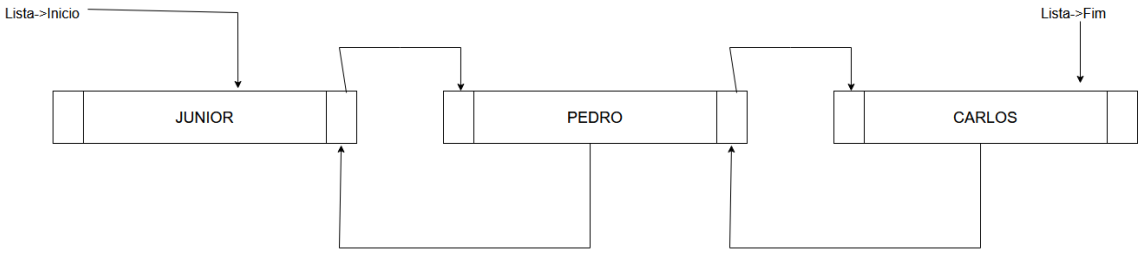
Função de Hash:

```
int h = 0;
for (int i = 0; i < s.length(); i++)
    h = (31 * h + s.charAt(i)) % M;
```

A função "funcaoHash()" calcula o índice da tabela hash com base no nome fornecido. Ela utiliza uma abordagem simples de soma de caracteres multiplicada por um fator constante e realiza o módulo pelo tamanho da tabela.

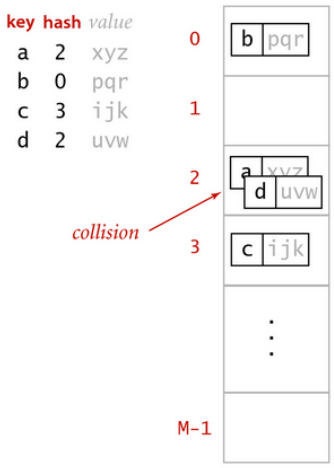
Inserção na Lista Encadeada:





A função "inserir_na_lista()" insere um nome na lista encadeada. Ela cria um novo nodo dinamicamente, preenche o nome e atualiza os ponteiros do nodo atual e do início da lista, se necessário.

Inserção na Tabela Hash:



A função "inserir()" insere um nome na tabela hash. Ela utiliza a função de hash para determinar o índice da tabela e chama a função "inserir_na_lista()" para adicionar o nome à lista correspondente.

Busca na Tabela Hash:

A função "busca()" utiliza a função de hash para determinar o índice da tabela e chama a função "buscar_na_lista()" para procurar o nome na lista correspondente. Ela imprime se o nome foi encontrado ou não.

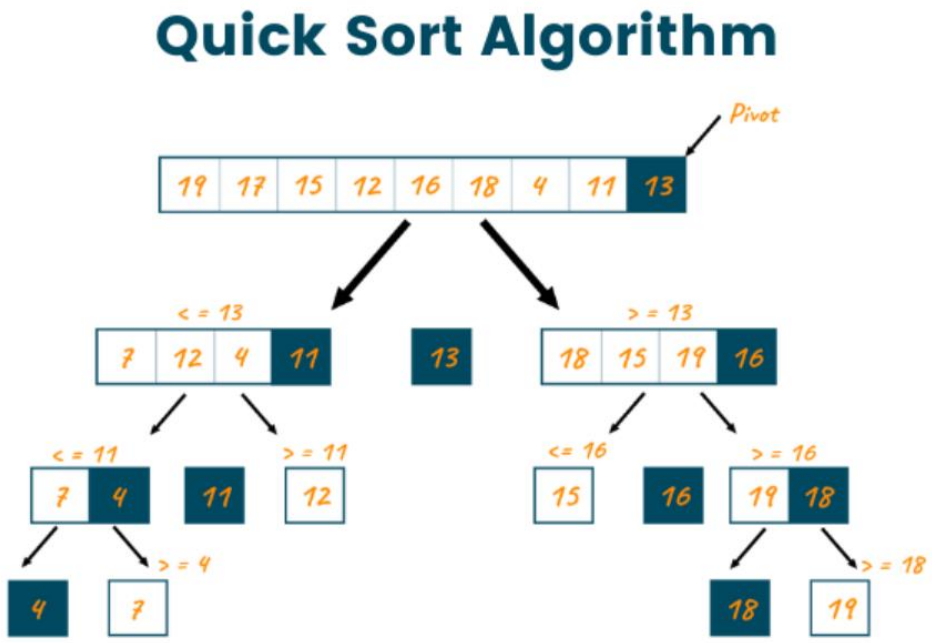
Impressão da Lista Encadeada:

A função "imprimir_lista()" percorre a lista encadeada e imprime os nomes armazenados nela.

Impressão da Tabela Hash:

A função "imprimirTabela()" percorre todas as listas da tabela hash e imprime os nomes armazenados em cada uma delas.

Ordenação da Lista Encadeada:



As funções do método Quick Sort são utilizadas para ordenar os elementos da lista encadeada com base no nome. Elas implementam o algoritmo de ordenação rápida para realizar a ordenação de forma eficiente. Foi utilizada a função string compare para auxiliar na comparação de strings.

Interface do Usuário:

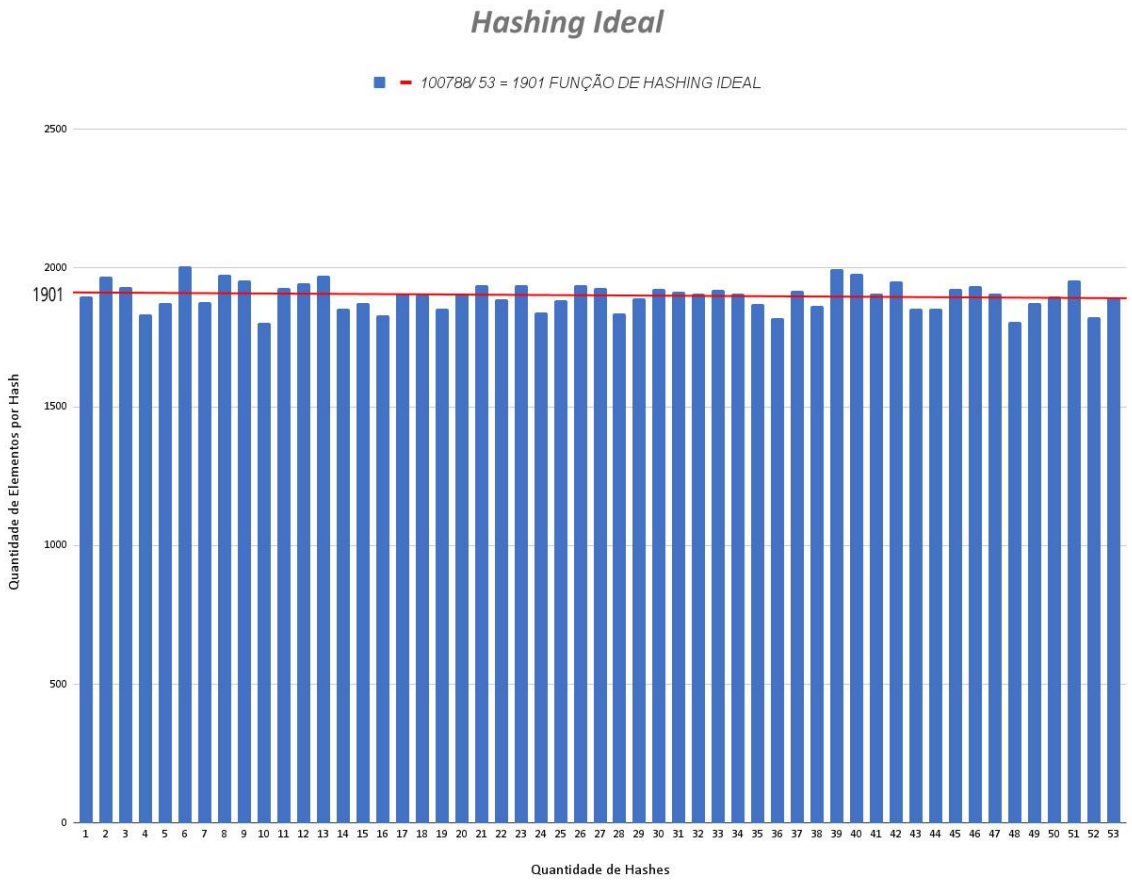
```
********** TABELA HASH **********
1. Inserir nome
2. Buscar nome
3. Imprimir listas
4. Ordenar listas
5. Exibir Histograma
6. Sair
*********************************
Escolha uma opтпo:
```

A função "main()" implementa uma interface de linha de comando para interagir com a tabela hash. Ela apresenta um menu com opções para inserir nomes, buscar nomes, imprimir listas, ordenar listas e exibir um histograma dos tamanhos das listas. Além disso, os nomes são lidos de um arquivo de texto chamado "nomes.txt" e são inseridos na tabela hash durante a inicialização.

Tratamento de Colisão:

No caso de colisões, quando diferentes chaves produzem o mesmo valor de hash, foi adotada a abordagem de encadeamento separado (chaining). Cada slot da tabela hash contém uma lista encadeada de elementos que mapeiam para o mesmo índice. Quando ocorre uma colisão, o novo elemento é adicionado à lista correspondente.

Análise Estatística:



Para avaliar a distribuição dos elementos na tabela hash, calculamos a média e o desvio padrão da quantidade de elementos por chave. A média foi calculada somando todos os valores e dividindo pela quantidade de chaves. Em seguida, calculamos a diferença entre cada valor e a média, elevamos ao quadrado essas diferenças e somamos todas elas. Dividimos esse resultado pelo número de chaves e tiramos a raiz quadrada para obter o desvio padrão. O desvio padrão resultante foi de aproximadamente 66,45, que é um resultado muito satisfatório levando em conta tamanho da tabela.

Conclusão:

A implementação da tabela hash com listas encadeadas mostrou-se eficiente para armazenar e recuperar informações. O tratamento de colisão por encadeamento separado permitiu lidar adequadamente com colisões e manter um desempenho satisfatório. A média e o desvio padrão da distribuição dos elementos indicam uma distribuição razoavelmente uniforme. A estrutura é capaz de lidar com muitos dados e proporcionar acesso rápido e eficiente às informações armazenadas.