# Trabalho prático de Algoritmos e Estruturas de Dados III Hashing

# Wilian Pereira dos Santos<sup>1</sup>

<sup>1</sup>Departamento de Informática - Universidade Federal do Paraná (UFPR)

Curitiba - PR - Brasil

# Introdução

O trabalho busca implementar tabelas hash de endereçamento aberto com tratamento de colisão feito em uma tabela extra, em linguagem C, e sintetizar o uso de algumas das funções usadas.

## Cálculo da posição nas tabelas

As tabelas hash dependem de funções que definem a posição ou index das chaves no vetor, e elas dependem de duas variáveis, **k** e **m**, onde: **k** é a chave a ser inserida/buscada e **m** é o tamanho da tabela.

A função para o cálculo do índice (index) da tabela 1 é: k % m;

A função para o cálculo do índice da tabela 2 é: Lm · (k · 0,9 - Lk · 0,9 J) J;

### **Funções**

A função "imprimeTabelas" é utilizada somente para confirmar se os testes estão corretos, e para tal, utiliza uma função que junta as duas tabelas em uma maior e a ordena, por meio de uma implementação recursiva do algoritmo quicksort, em ordem crescente de chaves.

A função "criaNo" retorna uma estrutura chamada "no", que contém uma chave inteira, e se a chave está presente na tabela ou se já foi removida.

A função "iniciaTabela" define a flag "removida" como -1 somente com o intuito de definir que o espaço alocado no vetor está livre, isto é, nenhuma chave foi inserida.

A função "busca" percorre as tabelas, por meio de funções específicas para cada tabela, para verificar se a chave escolhida está presente ou não. Lendo da esquerda para a direita, o fluxo que realiza a busca é:

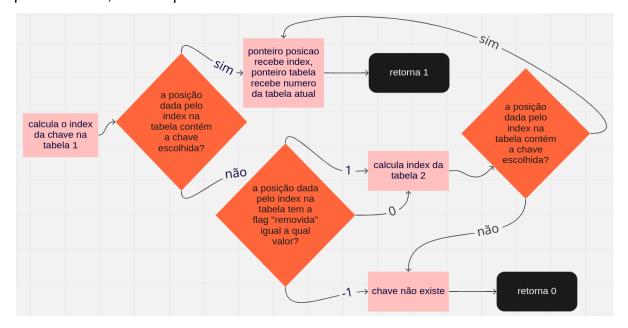


Figura 1: fluxograma de funcionamento da função de busca Fonte: elaborado pelo autor (2022)

A função "insere" verifica se já existe uma chave igual dentro das tabelas, caso exista, devido ao tratamento de inclusão fornecido no enunciado do trabalho, a função não faz nada. Caso não exista, ela calcula o index da primeira tabela e verifica se a posição está livre, se estiver, insere a chave e retorna 1. Caso a posição não esteja livre, ela calcula o index na segunda tabela, insere a chave e retorna 1. Caso aconteça algum erro, como falha na alocação de memória, é retornado 0.

Para se excluir uma chave, a função "busca" é instanciada e se a chave existir, a função "removeChave" é chamada. Essa função recebe como parâmetro os ponteiros da posição e da tabela obtidos através da função busca, então ela define a flag "removida" como 1 e retorna sucesso na operação.

#### Conclusão

Por mais elegante que seja utilizar hash, implementar as tabelas hash é uma tarefa simples, e com tempo de resposta sensacional, visto que a complexidade é O(1).