Atividade 11 - Tabelas Hash e Árvores Trie

19 de novembro de 2025

Problema A

```c
int buscar(TabelaHash* th, int valor) {
    // Implemente aqui a busca por um elemento na tabela Hash
    int indice = funcao_hash(valor, th->tamanho);

    No *aux = th->tabela[indice];

    while(aux != NULL){
        if(aux->valor == valor) return 1;

        aux = aux->prox;
    }
    return 0;
}
```

Problema B

```
int buscar(Trie* trie, const char* palavra) {
    // Implemente aqui a logica de buscar uma palavra
    NoTrie *atual = trie->raiz;

    if(!atual)
        return 0;

    int tamanho = strlen(palavra);

    for(int i = 0; i < tamanho; ++i){
        int pos = palavra[i] - 'a';

        // Palavra nao existe na trie
        if(pos >= TAMANHO_ALFABETO || pos < 0)
            return 0;
        if(atual->filhos[pos] == NULL)
            return 0;

        atual = atual->filhos[pos];
    }

    return atual != NULL && atual->fim_palavra;
}
```

Problema C

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct No {
    int valor;
    struct No * prox;
} No;

typedef struct TabelaHash {
    No** tabela;
    int tamanho;
} TabelaHash;

No* criaNo(int val){
    No *no = (No*) malloc (sizeof(No));
    no->valor = val;
    no->prox = NULL;
    return no;
}

TabelaHash* criaTabelaHash(int m){
    TabelaHash *ht = (TabelaHash*) malloc (sizeof(TabelaHash));
    ht->tabela = (No**) calloc (sizeof(No*), m);
    ht->tamanho = m;
    return ht;
}

int calculaIndice(int m, int val){
    return val % m;
}

void insereTabelaHash(TabelaHash *ht, int val){
    int indice = calculaIndice(ht->tamanho, val);
    No* novo_no = criaNo(val);
    novo_no->prox = NULL;

    No* aux = ht->tabela[indice];
    if(aux == NULL){
        ht->tabela[indice] = novo_no;
        return;
    }

    while(aux->prox != NULL){
        aux = aux->prox;
    }
    aux->prox = novo_no;
}

void imprimeTabelaHash(TabelaHash *ht){
    for(int i = 0; i < ht->tamanho; ++i){
        No* aux = ht->tabela[i];

        printf("%d", i);
        while(aux != NULL){
            printf(" -> %d", aux->valor);
            aux = aux->prox;
```

```
57          }
58          printf(" -> \\\n");
59      }
60  }
61
62  void liberaTabelaHash(TabelaHash *ht){
63      for(int i = 0; i < ht->tamanho; ++i){
64          No *aux = ht->tabela[i];
65
66          while(aux != NULL){
67              ht->tabela[i] = ht->tabela[i]->prox;
68              free(aux);
69              aux = ht->tabela[i];
70          }
71      }
72      free(ht->tabela);
73      free(ht);
74  }
75
76  int main(){
77      int n; scanf("%d", &n);
78
79      while(n--){
80          int m, c; scanf("%d %d", &m, &c);
81          TabelaHash *ht = criaTabelaHash(m);
82
83          for(int i = 0; i < c; ++i){
84              int x; scanf("%d", &x);
85
86              insereTabelaHash(ht, x);
87          }
88
89          imprimeTabelaHash(ht);
90          liberaTabelaHash(ht);
91          if(n > 0) printf("\n");
92      }
93
94      return 0;
95  }
```