SERVIÇO PÚBLICO FEDERAL · MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE VIÇOSA · UFV
CAMPUS FLORESTAL

# Trabalho 1 - AEDS 1

## Dicionário implementado por lista encadeada

Aymê Faustino [4704]
Vitor Hugo França [4684]

Florestal - MG
2022

**Sumário**

## 1. Introdução

Neste trabalho foi proposto que fosse implementado listas lineares e TADs para gerenciar a ocorrência de palavras em um texto. O objetivo era montar um TAD Dicionário contendo listas das palavras que o texto possui e que começam com cada letra do alfabeto e para cada palavra deveria ser criada uma lista informado a(s) linha(s) do texto na(s) qual(is) ela aparece. Um programa de testes deverá ser implementado, de maneira a receber um texto qualquer, criar o TAD Dicionário para o mesmo e, em seguida, permitir que determinadas operações desse TAD possam ser executadas por um usuário. A estrutura de dados a ser utilizada para implementação das listas lineares será lista simplesmente encadeada.

## 2. Organização

Na Figura 1 é possível visualizar a organização do projeto. Na pasta **TP1—AEDS**/ está a implementação do projeto, separada em módulos. As pastas **Dicionario/, ListaDePalavras/, Numeros/** e **Palavra/** contém os TADs solicitados, junto com a estrutura de dados, lista simplesmente encadeada .

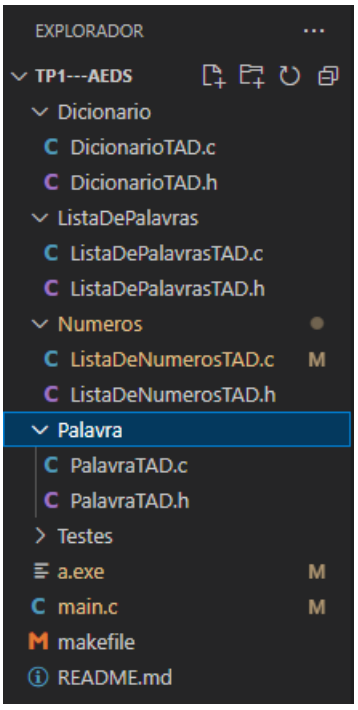


**Figura 1 - Repositório do projeto.**

Na pasta **Testes/** está o arquivo de texto que nos foi disponibilizado como exemplo e que foi usado para testes. Para executar o projeto, foi utilizado um arquivo **Makefile** com os comandos utilizados para compilar e executar os códigos.

## 3. Desenvolvimento

### 3.1 TAD Lista de números

Como foi sugerido no trabalho, achamos necessário a criação deste tad com o intuito de ser uma lista simplesmente encadeada de números inteiros, que representam as linhas nas quais uma palavra foi encontrada. A estrutura célula desse tad contém um inteiro num, que será o indicador da linha e um ponteiro do tipo CelulaNumeros que aponta para a próxima célula disponível. Este tad é uma implementação simples de lista encadeada sem o uso de célula cabeça. Foram feitas operações como inicialização da lista, verificação se a lista está vazia, inserção no final da lista, remoção no início da lista e por último uma função de impressão da lista.

### 3.2 TAD Palavra

O primeiro passo para a criação do TAD Palavra foi a criação do tipo de dados Palavra que contém internamente a estrutura que representa a palavra. Cada estrutura do tipo palavra possui uma string que representa a palavra em si e uma estrutura Lista de números, que será usada para representar a(s) linha(s) onde teve ocorrência daquela palavra. Foi necessário também importar o arquivo *ListaDeNumeros.h* onde está a estrutura lista de números inteiros. Neste TAD foram implementadas algumas funções que tinham sido solicitadas e outras que fomos precisando no decorrer do trabalho, mas que estão bem comentadas no código.

### 3.3 TAD Lista de palavras

Para a criação do TAD Lista de palavras foi necessário a criação do tipo de dados *ListaWord* para isso temos que importar o arquivo onde o TAD Palavra foi declarado que no caso é o *PalavraTAD.h* e criar a estrutura da célula da lista. Cada célula

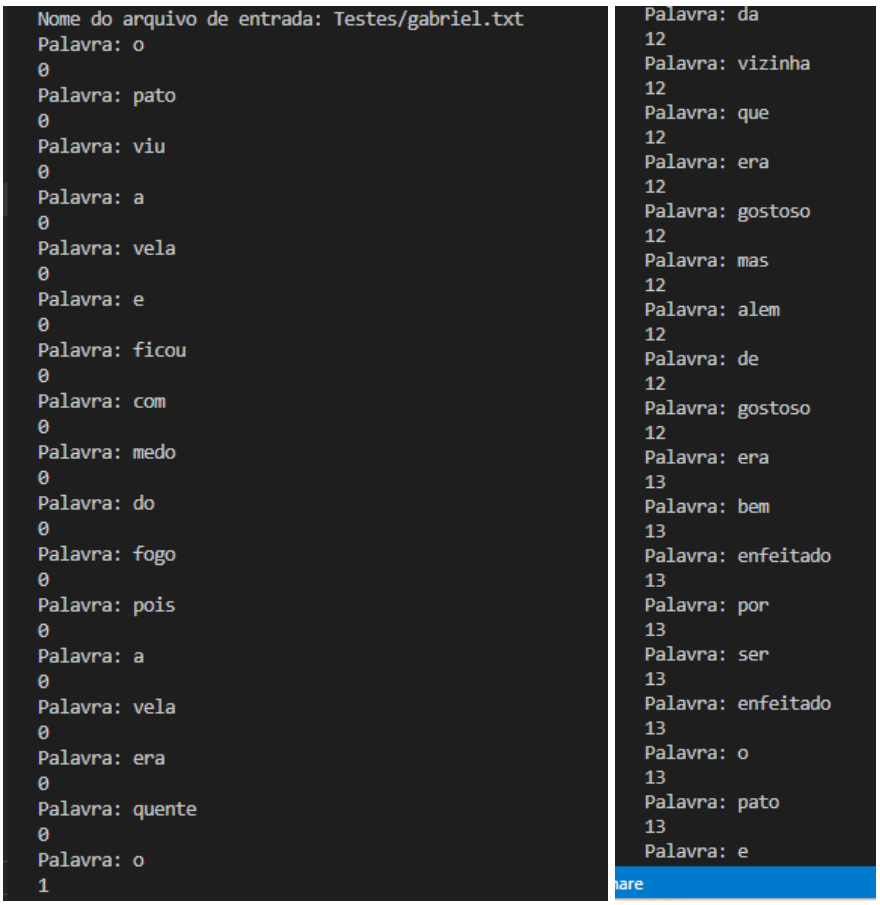
contém uma estrutura palavra e um ponteiro *pProx* do tipo *CelulaLista* que será usado para guardar o endereço da próxima célula. O próximo passo é a criação da estrutura da Lista de Palavras que contém, dois ponteiros do tipo *CelulaLista* o primeiro com o nome *primeiroLista* que aponta para a primeira célula da lista, o segundo com o nome *ultimoLista* que aponta para a última célula da lista e um int qtd que diz a quantidade de células que tem na lista.
No TAD Lista de Palavras foi implementada uma lista simplesmente encadeada com o uso de célula cabeça. As funções presentes neste TAD são::
*Cria_ListadPalavras* - que inicializa a lista com a célula cabeça, e posiciona os ponteiros.
*Insere_fim* - que insere uma nova palavra no fim da lista.
*Remove_palavra* - que passada a palavra por parâmetro, percorre a lista e quando encontra à remove.
*Remove_palavra_fim* - que remove a última palavra da lista.
*Verifica_palavra* - que passada a palavra, verifica se já existe na lista e retorna 0 se a palavra existir na lista.
*Retorna_qtd* - que retorna a quantidade de células que existe na lista.
Imprime (que imprime toda a lista de palavras) -
*ListaE_Vazia* - que nos diz quando a lista é composta somente pela célula cabeça.

### 3.4 TAD Dicionário

Para a criação TAD Dicionário, foi necessário importar o arquivo *ListaDePalavras.h* que contém a estrutura *ListaWord*. Na estrutura da célula do dicionário, temos uma estrutura do tipo *ListaWord*, um caractere que diz a letra que estara na lista, e um ponteiro *CelulaListaPalavras* que aponta para a proxima celula. Já a estrutura *Dicionarie* contém dois ponteiros do tipo *CelulaListaPalavras*, chamados *primeiroListaPalavras e ultimoListaPalavras* que vão marcar o início e o fim do Dicionário.

## 4. Resultados

De resultados obtivemos a lista de todas as palavras presentes no texto com suas respectivas linhas.




Nestas figuras estão os prints de como a lista está sendo imprimida.

## 5. Conclusão

Não conseguimos obter 100% o resultado esperado, pois nossa lista não ficou separada por letra e em ordem alfabética.Mas foi um trabalho proveitoso, pois treinamos e aprimoramos o funcionamento de um tipo abstrato de dados e a lista simplesmente encadeada.

## 6. Referências

[1] Github. Disponível em: <https://github.com/> Último acesso em: 15 de Outubro de 2022.

[2] Youtube. Disponível em <https://www.youtube.com/watch?v=I9etpvQvMPo> Ultimo acesso em: 08 de Outubro de 2022.

[3] Ziviani, N. Projeto de Algoritmos. Último acesso em: 15 de Outubro de 2022.