

**TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**DISCIPLINA DE ALGORITMOS – 2022.2**

IGOR GABRIEL DE ARAÚJO DANTAS

**TRABALHO DE VETORES DINÂMICOS**

ALOCAÇÃO DINÂMICA E LISTA LIGADA

**NATAL/RN**

09/01/2023

**SUMÁRIO**

1. Introdução
2. Vetores dinâmicos
3. Implementação
   1. Organização dos arquivos fontes
   2. Vetores com alocação dinâmica
   3. Lista ligada
   4. Testes
4. Resultados
5. Conclusão

**1. INTRODUÇÃO**

O principal objetivo do trabalho é a prática de gerenciamento de memória na linguagem de programação C, feito por meio da criação de duas bibliotecas: uma de alocação dinâmica e outra de lista duplamente ligada, cujos conceitos serão detalhados ao longo do relatório. Foram implementadas diversas funções em cada biblioteca e elaborados casos de teste, que também serão exibidos a seguir. Os resultados permitem que se possa visualizar, em casos específicos, quando um método se mostra mais eficaz em relação ao outro.

O ambiente de trabalho utilizado na implementação dos códigos para esse relatório foi o Visual Studio Code.

**2. VETORES DINÂMICOS**

Vetores dinâmicos são estruturas de dados de tamanho variável, iniciando com tamanho constante e aumentando o tamanho da estrutura quando necessário. Nesse tipo de organização, é possível gerenciar a quantidade de memória alocada para utilização. Existem diferentes formas de implementação desses vetores e, nesse trabalho, foram desenvolvidas duas bibliotecas.

A primeira biblioteca utiliza alocação dinâmica, que funciona com um ponteiro para um vetor estático, aumentando a capacidade do vetor quando ele está completo e um novo elemento precisa ser inserido.

A segunda biblioteca utiliza lista duplamente ligada, que consiste em um conjunto de registros sequencialmente ligados, chamados de nós, que armazenam um ponteiro para o anterior e outro para o próximo. O início e o fim do vetor são conhecidos e é possível percorrê-lo tanto do início para o fim, quanto no sentido contrário.

**3. IMPLEMENTAÇÃO**

A seguir, os testes elaborados e seus respectivos resultados serão exibidos, destacando o funcionamento de cada uma das bibliotecas criadas para esse trabalho.

**3.1. ORGANIZAÇÃO DOS ARQUIVOS FONTES**

Inicialmente, é importante demonstrar como os arquivos foram organizados. Ambas as implementações foram armazenadas em diretórios distintos, cada um contendo dois arquivos da biblioteca e um arquivo *main*, responsável pelos testes das funções, além dos arquivos executáveis gerados pelo compilador e um *.txt* que armazena o resultado do *main*.

Na parte de alocação dinâmica, a biblioteca é composta pelos arquivos *array\_list\_int.c* e *array\_list\_int.h*, que são utilizados pelo *main.c*, como demonstra a imagem a seguir:

Texto

Descrição gerada automaticamente

A biblioteca de lista duplamente ligada é organizada da mesma forma, a diferença está apenas no nome dos arquivos – *linked\_list\_int.c* e *linked\_list\_int.h* – como mostra a imagem:

Texto, Aplicativo

Descrição gerada automaticamente

Os códigos podem ser vistos detalhadamente nesses links: <https://github.com/igorrgabr/algoritmos/tree/main/projeto/alocação%20dinâmica>; <https://github.com/igorrgabr/algoritmos/tree/main/projeto/lista%20ligada>.

**3.2. VETORES COM ALOCAÇÃO DINÂMICA**

**3.3. LISTA LIGADA**

**3.4. TESTES**

**4. RESULTADOS**

**5. CONCLUSÃO**