Sumário

[1. Introdução: 2](#_Toc144310519)

[2. Implementação: 2](#_Toc144310520)

[3. Testes 2](#_Toc144310521)

[4. Conclusão 2](#_Toc144310522)

[Referências 2](#_Toc144310523)

[Anexos 2](#_Toc144310524)

[NomeArquivo.c 3](#_Toc144310525)

## Introdução:

*Neste trabalho prático, o objetivo é adaptar o algoritmo Quicksort em linguagem C para ordenar um vetor de palavras. O programa também deve contar o número de trocas e comparações realizadas, mostrar o vetor ordenado na janela do console e gerar um arquivo de saída formatado de acordo com as especificações fornecidas no enunciado.*

Xpto ...

#### GitHub:

*(Na seção de introdução, disponibilizar também o endereço compartilhado do código-fonte disponibilizado no GitHub).*

Xpto ...

## Implementação:

Estrutura de Dados e Variáveis

Foi utilizado um vetor de ponteiros para caracteres (char \*arr[20]) para armazenar as palavras a serem ordenadas.

Foram definidas variáveis para contagem do número de trocas e comparações.

Foram utilizadas as funções strcpy e strcmp da biblioteca string.h para manipulação de strings.

Funcionamento do Programa Principal

O programa principal implementa o algoritmo Quicksort para ordenar o vetor de palavras.

Durante a ordenação, as comparações e trocas são contadas conforme as palavras são comparadas e rearranjadas.

O vetor ordenado é exibido na janela do console.

Um arquivo de saída é gerado com o layout especificado no enunciado, incluindo o número de trocas, o número de comparações e o vetor ordenado.

Ambiente de Desenvolvimento

O código foi desenvolvido utilizando o compilador MinGW.

Para compilar e executar o programa, basta seguir as instruções do ambiente de desenvolvimento utilizado.Xpto ...

## Testes

*Aqui estão os resultados dos testes realizados:*

*Número de trocas: 15*

*Número de comparações: 64*

*Vetor ordenado:*

*abacate*

*abacaxi*

*amora*

*banana*

*caqui*

*cereja*

*figo*

*framboesa*

*goiaba*

*kiwi*

*laranja*

*maca*

*manga*

*melancia*

*morango*

*papaya*

*pera*

*pêssego*

*uva*

*Teste 2:*

*Número de trocas: 0*

*Número de comparações: 19*

*Vetor ordenado:*

*abacate*

*abacaxi*

*amora*

*banana*

*caqui*

*cereja*

*figo*

*framboesa*

*goiaba*

*kiwi*

*laranja*

*maca*

*manga*

*melancia*

*morango*

*papaya*

*pera*

*pêssego*

*uva*

## 

## Conclusão

*O trabalho foi concluído com sucesso, e o algoritmo Quicksort foi adaptado para ordenar um vetor de palavras. Os resultados dos testes indicam que o programa está funcionando corretamente, contando as trocas e comparações conforme esperado. As principais dificuldades encontradas estiveram relacionadas à manipulação de strings em C, mas essas dificuldades foram superadas com a ajuda das funções da biblioteca string.h.*

*Para futuras melhorias, pode-se considerar a implementação de um algoritmo de ordenação alternativo para fins de comparação de desempenho*

## Referências

*Documentação da biblioteca string.h*

*Chatgpt*

## Anexos

Atividade 02.c;

Output: atividade 02.exe, saída.txt.