Sumário

1. Introdução:

O objetivo deste trabalho é ler os dados de um vetor de palavras, processá-los e organiza-los em ordem crescente. Será utilizado o algoritmo de ordenação QuickSort.

GitHub:

https://github.com/arthursleite/Estrutura-Dados-UCB/tree/main/tp02

2. Implementação:

Foi utilizada a linguagem C com assistência da IDE Visual Studio Code e compilador GCC.
A implementação foi feita utilizando somente um arquivo .c contendo toda a lógica, funções e variáveis do programa.

3. Testes

Essas são as bibliotecas utilizadas e as variáveis globais que serão utilizadas:

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

int trocas = 0;
int comparacoes = 0;
```

Estas são as funções auxiliares que serão utilizadas:

```c
void swap(char **a, char **b)
{
    char *temp = *a;
    *a = *b;
    *b = temp;
    trocas++;
}

int partition(char *arr[], int low, int high)
{
    char *pivot = arr[high];
    int i = (low - 1);

    for (int j = low; j <= high - 1; j++)
    {
        if (strcmp(arr[j], pivot) <= 0)
        {
            i++;
            swap(&arr[i], &arr[j]);
        }
        comparacoes++;
    }
    swap(&arr[i + 1], &arr[high]);
    return (i + 1);
}

void quickSort(char *arr[], int low, int high)
{
    if (low < high)
    {

        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
```

Estão é a função principal:

```c
int main()
{
    char *arr[20] = {
        "maca", "banana", "pera", "uva", "laranja", "abacaxi", "limão", "manga", "abacate", "kiwi",
        "cereja", "morango", "pêssego", "goiaba", "melancia", "framboesa", "amora", "caqui", "figo", "papaya"};

    trocas = 0;
    comparacoes = 0;

    quickSort(arr, 0, 19);

    printf("Vetor ordenado:\n");
    for (int i = 0; i < 20; i++)
    {
        printf("(%d) %s\n", i + 1, arr[i]);
    }

    FILE *arquivoSaida = fopen("saida.txt", "w");
    if (arquivoSaida == NULL)
    {
        perror("Erro ao criar o arquivo de saída");
        return 1;
    }

    fprintf(arquivoSaida, "Número de trocas: %d\n", trocas);
    fprintf(arquivoSaida, "Número de comparações: %d\n", comparacoes);
    fprintf(arquivoSaida, "Vetor ordenado:\n");
    for (int i = 0; i < 20; i++)
    {
        fprintf(arquivoSaida, "(%d) %s\n", i + 1, arr[i]);
    }
    fprintf(arquivoSaida, "Mediana: %s e %s\n", arr[10], arr[11]);

    fclose(arquivoSaida);

    return 0;
}
```

4. Conclusão

O programa funcionou perfeitamente de acordo com o que foi pedido no enunciado.

Referências

https://www.geeksforgeeks.org/quick-sort/

https://www.todamateria.com.br/media-moda-e-mediana/

Anexos

tp02.c

saida.txt