Universidade Católica de Brasília

Pró-Reitoria Acadêmica
Curso de Bacharelado em Ciência da Computação
Trabalho da Disciplina de Estrutura de Dados

Documento de Especificação

Autor(es): Diego Santos Silva Ferreira, Gleiciany Oliveira Pereira, João Lucas Dutra Gomes, Letícia Rodrigues de Sousa, Maria Vitória Ferreira Lopes, Mateus Vitor Venâncio de Deus.
Orientador: Prof. Zoé Roberto Magalhães Júnior.

DIEGO SANTOS SILVA FERREIRA
GLEICIANY OLIVEIRA PEREIRA
JOÃO LUCAS DUTRA GOMES
LETÍCIA RODRIGUES DE SOUSA
MARIA VITÓRIA FERREIRA LOPES
MATEUS VITOR VENÂNCIO DE DEUS

DOCUMENTO DE ESPECIFICAÇÃO

Documento apresentado ao Curso de graduação de Bacharelado em Ciência da Computação da Universidade Católica de Brasília, como requisito parcial para obtenção da aprovação na disciplina de Estrutura de Dados

Brasília
2025

RESUMO

Referência: DEUS, Mateus Vitor Venâncio de. Ferreira, Diego Santos Silva. GOMES, João Lucas Dutra. LOPES, Maria Vitória Ferreira. SOUSA, Letícia Rodrigues de. PEREIRA, Gleiciany Oliveira. Documento de Especificação, 2025. nr p. Bacharelado em Ciência da Computação – UCB – Universidade Católica de Brasília, Taguatinga – DF, 2024.

Este projeto tem como finalidade o desenvolvimento de uma aplicação em linguagem C para leitura, ordenação e exportação de dados de funcionários contidos em arquivos no formato CSV. O programa oferece ao usuário a opção de ordenar os registros com base em três critérios distintos: nome, setor ou idade. Para isso, utiliza-se o algoritmo de ordenação Bubble Sort, apropriado para conjuntos de dados pequenos a médios. A aplicação realiza o tratamento adequado de entrada de dados e de alocação dinâmica de memória, garantindo maior robustez e segurança na manipulação das informações. Os dados ordenados são exibidos no terminal e gravados em um novo arquivo CSV, conforme o nome informado pelo usuário. Trata-se de uma solução prática, leve e eficiente para a organização de cadastros de funcionários, com potencial de expansão para funcionalidades mais avançadas.

Palavras-chave: Programação em C; CSV; Ordenação de dados.

ABSTRACT

This project aims to develop an application in the C programming language to read, sort, and export employee data stored in CSV files. The program allows the user to choose one of three sorting criteria: name, department, or age. It employs the Bubble Sort algorithm, which is suitable for small to medium datasets. The application handles user input and dynamic memory allocation properly, ensuring robustness and safety in data processing. Once sorted, the records are displayed in the terminal and saved in a new CSV file specified by the user. This is a practical, lightweight, and efficient solution for organizing employee records, with potential for future enhancements and added features.

Keywords: C programming; CSV; data sorting.

SUMÁRIO

1. INTRODUÇÃO

O processamento eficiente de dados é uma demanda constante em sistemas computacionais, especialmente no contexto da gestão de informações administrativas. Neste cenário, arquivos no formato CSV (Comma-Separated Values) são amplamente utilizados por sua simplicidade e compatibilidade com diversas aplicações. Este projeto propõe o desenvolvimento de uma aplicação em linguagem C capaz de ler, ordenar e exportar dados de funcionários contidos em arquivos CSV. Através de uma interface simples via terminal, o usuário pode selecionar o critério de ordenação — nome, setor ou idade — e obter como resultado um conjunto de dados organizado, exibido na tela e salvo em um novo arquivo. A escolha da linguagem C se justifica por sua eficiência em manipulação de memória, leitura de arquivos e controle de baixo nível, o que proporciona uma aplicação leve, estável e de fácil manutenção. Este trabalho demonstra, portanto, como é possível integrar conceitos fundamentais de programação estruturada com soluções práticas para o tratamento e organização de dados.

1.1 MOTIVAÇÃO

A motivação para o desenvolvimento deste programa surgiu da necessidade comum em empresas e instituições de manter seus registros atualizados e organizados. Muitas vezes, os dados são armazenados em planilhas simples ou arquivos de texto, e a ordenação ou filtragem dessas informações manualmente pode se tornar uma tarefa trabalhosa e suscetível a erros. Ao utilizar uma linguagem de baixo nível como C, é possível explorar o controle de memória, desempenho e manipulação direta de arquivos, criando uma ferramenta eficiente, leve e adequada para sistemas com recursos limitados ou que necessitem de soluções embarcadas.

2. OBJETIVOS DA SOLUÇÃO

Este documento descreve a solução proposta para a implementação de um sistema em linguagem C com o objetivo de automatizar a leitura, ordenação e exportação de dados de funcionários armazenados em arquivos no formato CSV. A aplicação permite ao usuário ordenar os dados segundo três critérios específicos: nome, setor ou idade. O sistema é projetado para facilitar o gerenciamento de informações cadastrais, oferecendo uma solução simples e eficiente para manipulação de dados em pequena escala.

Além disso, a aplicação possui a capacidade de criar um novo arquivo CSV contendo os dados ordenados, proporcionando uma maneira prática de trabalhar com registros de funcionários sem depender de ferramentas de planilhas ou outros softwares mais complexos.

3. ENTRADA E SAÍDA DE DADOS

3.1 ENTRADA DE DADOS

A entrada de dados para a aplicação é realizada por meio de um arquivo CSV, que deve estar formatado da seguinte maneira:

- Cada linha do arquivo contém três campos separados por vírgula:
  nome,setor,idade
- Exemplo de entrada válida:
  Ana Maria,A,28
  João Silva,B,35
  Carlos Souza,C,42

3.2 ENTRADA DO USUÁRIO VIA TERMINAL

A aplicação solicita ao usuário as seguintes informações:

- **Critério de ordenação:**
  1. **Ordenar por nome.**
  2. **Ordenar por setor.**
  3. **Ordenar por idade.**
- **Nome do Novo Arquivo CSV**: Após a ordenação dos dados, o usuário será solicitado a fornecer o nome para o arquivo CSV onde os dados ordenados serão salvos.

3.3 SAÍDA DE DADOS

- A aplicação imprime no terminal os registros ordenados de acordo com o critério selecionado:
  Nome | Setor: x | Idade: y
- A aplicação cria um novo arquivo CSV com o nome fornecido pelo usuário, contendo os dados de funcionários ordenados no formato:
  nome, setor, idade

4. ALGORITMO DE ORDENAÇÃO UTILIZADO

4.1 BUBBLE SORT

A solução implementa o algoritmo **Bubble Sort** para realizar a ordenação dos dados. O Bubble Sort é um algoritmo de ordenação simples e eficiente para pequenas quantidades de dados, como o escopo proposto neste projeto. Ele funciona da seguinte maneira:

1. **Comparação e Troca**: O algoritmo percorre a lista de registros e compara elementos adjacentes. Se os elementos estiverem fora de ordem, eles são trocados de posição.
2. **Repetição**: Esse processo é repetido até que a lista esteja completamente ordenada.

- **Critérios de Ordenação:**
- **Por nome**: Utiliza a função **strcmp** para comparar strings.
- **Por setor**: Compara os caracteres que representam o setor.
- **Por idade**: Compara os valores inteiros das idades.

5. ESTRUTURA DO CÓDIGO E PRINCIPAIS FUNÇÕES IMPLEMENTADAS

A aplicação é organizada em funções modulares que facilitam a leitura, manutenção e expansão do código. A seguir, são descritas as principais funções do código:

5.1 ESTRUTURA PRINCIPAL (STRUCT)

A aplicação utiliza uma struct chamada **Funcionario** para armazenar os dados dos funcionários.

```
typedef struct {
   char nome[100];
   char setor;
   int idade;
} Funcionario;
```

5.2 FUNÇÕES IMPLEMENTADAS

- **main( ):** Função principal que controla o fluxo do programa.
- **lerArquivo( ):** Função que lê os dados do arquivo CSV e armazena em memória.

- **ordenar( ):** Função que chama a função de ordenação escolhida pelo usuário.
- **sortNome( ):** Ordena os dados por nome utilizando o algoritmo Bubble Sort.
- **sortSetor( ):** Ordena os dados por setor utilizando o algoritmo Bubble Sort.
- **sortIdade( ):** Ordena os dados por idade utilizando o algoritmo Bubble Sort.
- **troca( ):** Função que realiza a troca de dois elementos na lista.
- **imprimirFuncionarios( ):** Exibe os dados ordenados no terminal.
- **salvarArquivoNovo( ):** Grava os dados ordenados em um novo arquivo CSV.

6. PROBLEMAS ENCONTRADOS E SOLUÇÕES ADOTADAS

PROBLEMA	SOLUÇÃO IMPLEMENTADA
Entrada de dados errada (linhas em formato incorreto)	A função **sscanf( )** foi utilizada para validar a leitura dos dados, com mensagens de erro para entradas inválidas.
Falha ao ler arquivos grandes (falta de memória)	A alocação de memória foi tratada utilizando **malloc** e **realloc**, garantindo a expansão dinâmica da memória conforme necessário.
Erros ao manipular caracteres especiais	O código foi projetado para garantir a correta leitura e escrita de caracteres no formato UTF-8.
Mistura de entradas entre scanf e fgets	A função **getchar()** foi adicionada para limpar o buffer de entrada antes de usar **fgets().**

7. CONCLUSÃO

A aplicação proposta atendeu aos objetivos iniciais de leitura, ordenação e exportação de dados de funcionários de forma simples e eficiente. A escolha da linguagem C, com sua forte capacidade de manipulação de memória e arquivos, mostrou-se adequada para o tipo de solução desejada, permitindo a criação de uma ferramenta robusta e leve.

A implementação do algoritmo de ordenação Bubble Sort, embora simples, foi suficiente para atender ao problema, dado o pequeno porte dos dados manipulados. Além disso, a solução desenvolvida é facilmente extensível, permitindo que, no futuro, novas funcionalidades possam ser agregadas, como novos critérios de ordenação e filtragem de dados.