Instituto Superior de Engenharia

Politécnico de Coimbra

Programação

Metro Mondego

Pedro Emanuel Dinis Serrano nº 2016017926 - LEICE

Índice

1. Introdução

Este trabalho foi feito no âmbito da unidade curricular de Programação, no ano letivo 2022/2023. Foi feito totalmente usando CLion e respeita a norma C99, tendo como objetivo simular, a um nível básico, um programa de gestão das linhas do Metro Mondego.
Para tal, foram usados conceitos da linguagem C, tais como estruturas dinâmicas, estruturas ligadas e leitura e escrita em ficheiros de texto e binários.

2. Código fonte

O código fonte encontra-se dividido em quatro ficheiros de código (extensão .c) e três ficheiros *header* (.h):

- main.c
    - Contém os vários menus disponíveis para o utilizador navegar entre as várias opções das linhas e das paragens. É neste ficheiro que se encontra a maioria do código com o qual o utilizador interage.
- utils.c/.h
    - Contém funções para guardar e carregar as estruturas dinâmicas do programa, assim como uma função de leitura de linhas através de um ficheiro de texto e uma função para remover espaços de uma *string*.
- linhas.c/.h
    - Contém todas as funções relativas à manipulação da estrutura las linhas, tais como: criar/eliminar uma linha, adicionar/remover paragens de uma linha, entre outros.

- paragens.c/.h
    - Contém todas as funções relativas à manipulação da estrutura das paragens, tais como adicionar paragens, remover paragens, entre outros.

3. Estrutura de dados

Foram utilizadas três estruturas diferentes neste trabalho, as quais estão descritas abaixo:

- Paragem

```
typedef struct {
    char nome[100];
    char codigo[5];
} Paragem, *pParagem;
```

Esta estrutura é utilizada para guardar a informação acerca de cada paragem no *array* dinâmico de paragens válidas a serem utilizadas pelo resto do programa.

- Linha

```
struct linha{
    char nome[100];
    int totalP;
    pLinha prox;
    pParLinha paragens;
};
```

Esta estrutura é utilizada para guardar a informação das várias linhas. Contém, também, um ponteiro ParLinha que aponta para a primeira paragem da lista de paragens dessa linha.

- ParLinha

```
struct paragem{
    pParagem paragem;
    pParLinha prox;
};
```

Esta estrutura serve para representar uma paragem de uma linha. Contém um ponteiro para uma paragem do *array* de paragens válidas e um ponteiro para a próxima paragem dessa linha.

4. Estruturas dinâmicas

Foram implementados dois tipos de estruturas dinâmicas: um *array* dinâmico de paragens, tal como foi mencionado num enunciado, e uma lista de listas para a gestão de toda a informação das linhas. Cada nó da lista de linhas contém um ponteiro para a primeira paragem de cada linha. Decidi fazer esta implementação por parecer a mais simples de implementar e lidar em termos de adição e eliminação de paragens e linhas.