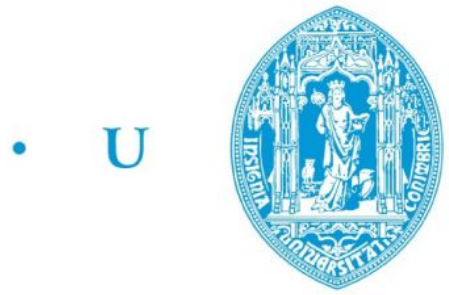


FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Princípios de Programação Procedimental

2018/2019 - 2º Semestre

Projeto Planeamento de Viagens no DEI

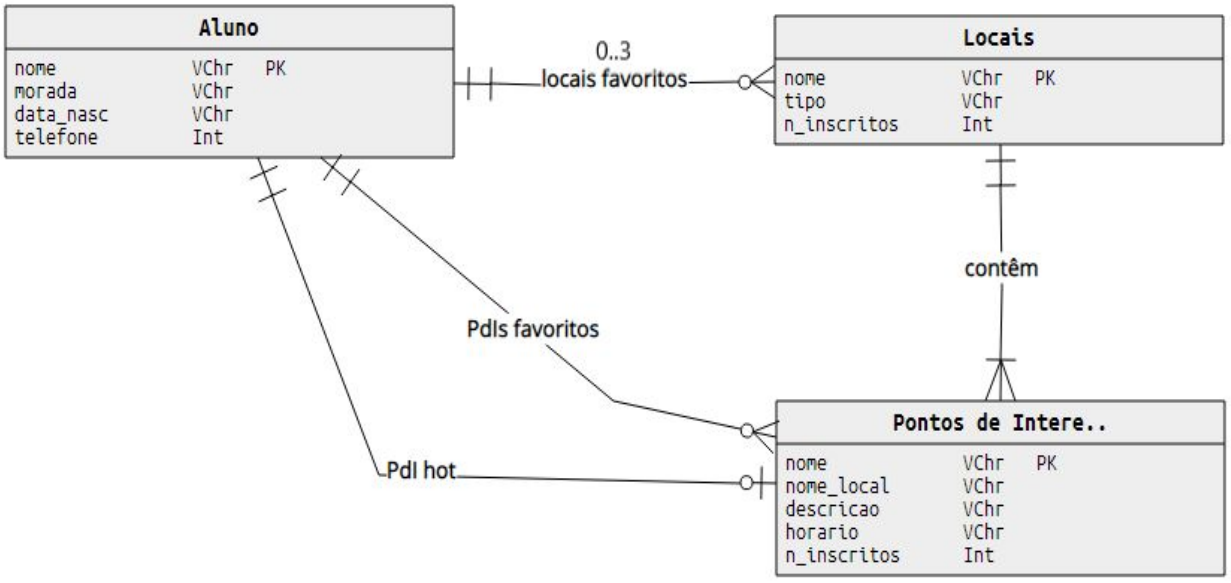
Alexandre Américo Ferreira

2016240875 – alexandre@student.dei.uc.pt– PL4

Guilherme Moita Pontes

2016242153 – pontes@student.dei.uc.pt – PL4

Representação da estrutura geral do programa



Estruturas:

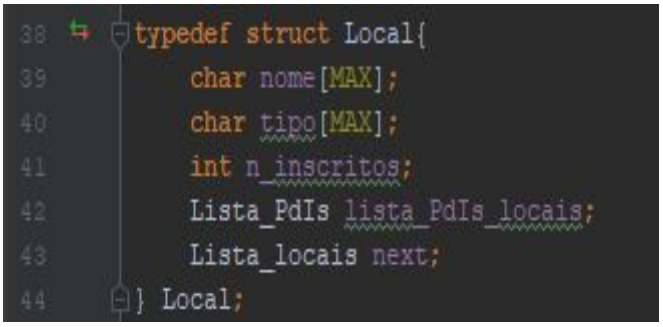
Após a análise do enunciado do trabalho proposto pelo docente, decidimos implementar 3 estruturas:

- **Aluno:** estrutura que armazena toda a informação pessoal do utilizador, o número de locais em que está inscrito, o nome do ponto de interesse e do local "hot" do utilizador, uma lista ligada de locais que corresponde aos locais que o user tem adicionados aos favoritos, duas listas de pontos de interesse, uma com os favoritos e outra que servirá para guardar os pontos de interesse da viagem que o programa gera para o utilizador, e, por fim, uma referência para o próximo aluno na lista ligada de alunos.

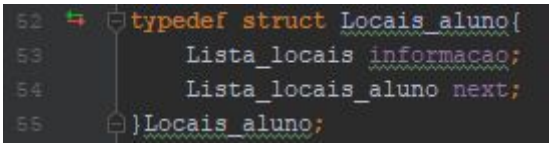
```
typedef struct Aluno{
    char nome[MAX];
    char morada[MAX];
    char data_nasc[MAX];
    int telefone;
    int n_locais;
    char PdI_hot[MAX];
    char local_hot[MAX];
    Lista_locais_aluno lista_locais_alunos;
    Lista_pdis_aluno lista_PdIs_alunos;
    Lista_pdis_aluno viagem;
    Lista_alunos next;
} Aluno;
```

- **Pontos de Interesse (PDI) :** estrutura que tem como objetivo armazenar todas as informações de um ponto de interesse tais como o nome, o nome do local (cidade, vila, ...) a que pertence, o número de utilizadores inscritos (ou seja, que definiram este ponto de interesse como favorito), a descrição do PDI, o seu horário e uma referência para o próximo ponto de interesse na lista.

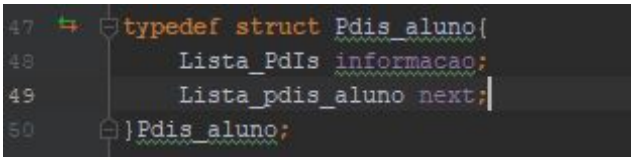
- **Local:** estrutura com a finalidade de guardar o nome do local, o tipo, o número de utilizadores inscritos (ou seja, que definiram este local como favorito) , uma lista de pontos de interesse com os pontos de interesse do respetivo local e um ponteiro do tipo Lista_Locais que referencia o próximo elemento da lista ligada de locais.



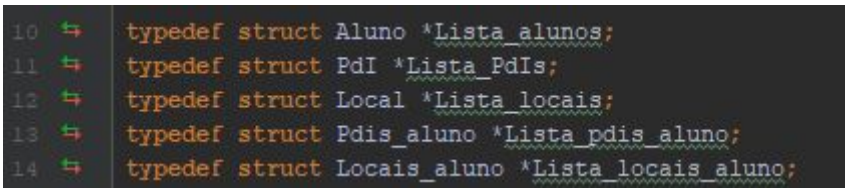
- **Locais_aluno:** estrutura com a finalidade de guardar um ponteiro para um local da lista de locais onde irá armazenar a informação do local e um atributo next que irá representar o próximo local da lista de locais do aluno.



- **Pdis_aluno:** estrutura com a finalidade de guardar um ponteiro para um ponto de interesse onde irá armazenar a informação do mesmo e um atributo next que irá representar o próximo local da lista de pontos de interesse do aluno.



- Estas estruturas são criadas recorrendo a ponteiros, da seguinte forma:



Funcionamento do Programa

Pode-se considerar que o programa se divide em 3 partes: tratamento de ficheiros, funções que realizam operações em listas ligadas e interface, que iremos detalhar de seguida.

Tratamento dos ficheiros

Existem dois tipos de funções relativas a ficheiros: carregamento de dados, usado ao iniciar o programa e gravação de dados, usado ao sair do programa para guardar o estado atual.

Os ficheiros são de texto, “locais.txt” para guardar locais e pontos de interesse e “alunos.txt” para guardar a informação relativa a alunos e eventuais locais e/ou pontos de interesse que tenham adicionados como favoritos. Nestes ficheiros, a informação está guardada linha a linha e uma linha em branco representa o fim da informação relativa aquele nó e o começo do próximo nó.

Interface

A interface do programa permite ao utilizador navegar pelo programa e realizar as várias operações que este está preparado para executar. Ela é formada por 4 menus:

1. **Menu Inicial:** menu com apenas três funcionalidades. Neste menu é possível fazer o registo de um novo utilizador caso ainda não esteja registado no programa, fazer login e por último fechar o programa. No registo, é pedido ao utilizador que preencha os vários atributos que permitem a criação de um novo aluno. Para aceder aos restantes menus, o utilizador tem que fazer login onde a credencial necessária para suceder nesta operação é o nome com que se registou. Por último, a opção de sair faz com que o programa acabe guardando em ficheiros de texto o estado atual da informação inserida.
2. **Menu Principal:** menu que apresenta as principais funcionalidades do programa. Nele é possível aceder ao menu de locais e de pontos de interesse, como também editar dados do aluno "logado", gerar a viagem automaticamente e visualizar todos os locais e pontos de interesse disponíveis e as estatísticas relativas à viagem gerada.
3. **Menu Locais:** Permite adicionar ou remover locais existentes no programa aos locais favoritos do aluno "logado".
4. **Menu Pontos de Interesse:** Mesma função que o menu de locais mas referente aos pontos de interesse. Neste menu, é possível ainda definir o ponto de interesse "hot" do utilizador, caso esse ponto de interesse já esteja na sua lista de favoritos.

Funções listas ligadas

As funções relativas à manipulação de listas ligadas dividem-se em:

1. **Criar listas:** o objetivo deste tipo de funções é criar o primeiro elemento para a lista ("header") alocando memória para um nó e definindo os seus atributos a NULL.
2. **Inserir:** esta função tem como objetivo inserir nós por ordem alfabética na lista passada como parâmetro.
    a. Nas funções de inserir alunos e locais apenas são passados por parâmetros a respectiva lista e os atributos necessários para criar um aluno/local. Depois de criado o nó, a lista é percorrida até ser encontrado o local onde deve ser inserido de maneira a que fique por ordem alfabética, caso não exista ainda na lista.
    b. Para as listas de pontos de interesse é necessário acrescentar o local a que este pertence por parametro, visto que consideramos que um ponto de interesse está sempre associado a um local. Assim sendo, depois de encontrado o local na lista de locais, o processo de inserção é semelhante ao anteriormente descrito.
    c. Por último, temos duas funções de inserção diferentes: adicionar locais e pontos de interesse ao nó aluno. A diferença nestas funções está no

facto de não se querer realizar uma cópia do local/ponto de interesse mas sim criar uma referência na lista de favoritos de aluno para esse local/ponto de interesse em questão.

3. **Procurar:** percorre uma lista ligada do início para o fim guardando o nó atual e o nó anterior. Caso encontre o elemento retorna 1, caso contrário retorna 0. Os ponteiros para os nós anterior e atual ficam disponíveis a ser usados por outras funções.
4. **Pesquisa:** utiliza a função procura acima descrita para retornar o nó procurado, caso exista.
5. **Destrói:** percorre a lista passado por parâmetro libertando memória para todos os nó.
6. **Imprime:** mostra na consola todos os nós da lista passada por argumento.
7. **Função gerar viagem:** tem como objetivo gerar uma viagem de recomendação consoante os locais e pontos de interesse favoritos do utilizador "logado", bem como o seu ponto de interesse "hot". Primeiramente, o método verifica se o utilizador possui um ponto de interesse "hot". Caso exista e o pdi hot esteja num dos locais favoritos, são inseridos na lista de viagem do user o pdi hot e outros 2 pontos de interesse. Segundamente, é percorrida a lista de locais do aluno e se o local para onde esta está a apontar for diferente do local hot definido anteriormente, percorremos a lista de pontos de interesse favoritos do utilizador para ver se algum destes pertence a um dos locais da lista do aluno. Assim damos prioridade aos pontos de interesse favoritos do user. No final de cada verificação, caso o número de locais adicionados à viagem seja inferior a 3, são adicionados pontos de interesse do local, ordenados por popularidade (utilizamos como critério para definir popularidade o número de inscritos no ponto de interesse), até que o número de locais seja igual a 3.
8. **Funções estatísticas da viagem:** gera uma avaliação da viagem numa escala de 0 a 3. Para fazer a avaliação da mesma são utilizadas 3 funções: **1- percentagem_pdi_hot**, responsável por contar o número de utilizadores que têm o pdi hot, do utilizador em questão, na sua viagem; **2 - percentagem_favoritos**, calcula quantos utilizadores possuem pontos de interesse na sua viagem que pertence à lista de pontos de interesse favoritos do user; **3 - percentagem_pdis**, devolve o número total de inscritos nos pontos de interesse incluídos na viagem do utilizador. Depois, os dois primeiros valores são divididos pelo número de número de utilizadores (determinado pela função auxiliar get_n_utilizadores) e o último pelo número total de inscritos em todos os pdis (determinado pela função auxiliar get_n_inscritos). No final soma-se as 3 componentes obtendo o resultado final.