1 2 9 0 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

21/22

Manual do Programador

PRINCÍPIOS DE PROGRAMAÇÃO PROCEDIMENTAL

FRANCISCO QUERIDO - 2021221158 | MIGUEL BRAGA - 2021221519

DOCENTE: NUNO PIMENTA

Índice

1 - Objetivo do Projeto:

O objetivo deste projeto de PPP é criar uma aplicação para gerir as contas do bar dos alunos de uma escola. Esta aplicação deve possuir uma série de funcionalidades de forma que toda a informação sobre os alunos possa ser atualizada, adicionada e removida. Para além de manipulação de dados pessoais é ainda pedido que a aplicação mantenha registo das despesas dos mesmos.

A aplicação contém também vários tipos de proteções, tanto para o utilizador caso ele se engane a inserir os dados, como avisos caso o programa não consiga realizar alguma tarefa com sucesso (abrir um ficheiro ou ler uma informação, por exemplo).

2 - Funcionamento da Aplicação:

- 2.1 - main.c:

O programa inicializa (como indicado na consola) e carrega a informação dos ficheiros de texto para listas de forma que esta possa ser manipulada. De seguida, pede a data do dia para que a possa guardar como informação no ficheiro das despesas mais tarde.

De seguida, o programa entra num loop no qual vai estar permanentemente inserido até ser encerrado. Este loop apresenta um texto que pede ao utilizador que escreva uma de várias funcionalidades. Ao detetar que uma das funções foi escolhida, este entra de imediato dentro de várias condições que pedem mais informação e confirmações ao utilizador para poder executar a funcionalidade escolhida (ou seja, pede os parâmetros da função a executar).

Após recolher toda a informação necessária para que possa realizar a tarefa, o programa chama a função e fornece os devidos parâmetros.

Assim que a execução da funcionalidade se encontra finalizada o loop volta ao início perguntando de novo ao utilizador que funcionalidade pretende escolher.

Para terminar o programa basta escrever “Terminar” na consola. Ao teclar este comando, o programa guarda a informação que contém nas listas de alunos e de despesas nos respetivos ficheiros de texto.

- 2.2 - listas.c:

Este ambiente é o lugar onde se encontram todas as funções desta aplicação:

"**cria**" -> Cria um nó da lista de alunos. Com uma estrutura Aluno e um ponteiro para um NoAluno. Quando termina devolve o ponteiro para o nó criado (cabeçalho);

"**cria_saldo**" -> Cria um nó da lista de ordenamento de saldos dos alunos. Com um ponteiro para um NoAluno- que corresponde ao aluno em questão e outro para um NoSaldo-que corresponde ao nó seguinte da lista. Quando termina devolve o ponteiro para o nó criado (cabeçalho);

"**cria_despesas**" -> Cria um nó da lista de despesas. Com uma estrutura Despesa e um ponteiro para um NoDespesa. Quando termina devolve o ponteiro para o nó criado (cabeçalho);

"**le_ficheiro_alunos**" -> Lê a informação guardada no ficheiro "Alunos" e guarda-a na lista, lista_alunos. Esta função lê o ficheiro linha a linha sendo que recolhe a informação de cada aluno e recorre depois à função "**insere**" para guardar a informação na lista;

"**le_ficheiro_despesas**" -> Lê a informação guardada no ficheiro "Despesas" e guarda-a na lista, lista_despesas. Esta função lê o ficheiro linha a linha sendo que recolhe a informação de cada despesa e recorre depois à função "**insere_despesas**" para guardar a informação na lista;

"**insere**" -> Adiciona um NoAluno à lista, lista_alunos. Recorre à função "**ordena**" para ordenar a lista por ordem alfabética;

"**insere_saldo**" -> Adiciona um NoSaldo à lista, lista_saldo. Recorre à função "**procura_saldo**" para obter os ponteiros para nós da lista, lista_alunos com saldo abaixo do especificado, posteriormente ordenando o NoSaldo adicionado por ordem decrescente de saldo;

"**insere_despesas**" -> Adiciona um NoDespesa à lista, lista_despesas. Recorre à função "**procura**" para obter nós do aluno que efetuou a despesa e de seguida à função "**ordena_despesas**" para ordenar o NoDespesa adicionado na lista de despesas, por ordem crescente de números de alunos;

"**carregar**" -> Adiciona um determinado valor a carregar ao saldo do aluno pretendido. Recorre à função procura para obter os nós do aluno em questão;

"**elimina**" -> Elimina um aluno. Recorrendo à função "**procura**" para obter os nós do aluno em questão e posteriormente libertando a memória usada por esse nó;

"**imprime**" -> Apresenta os dados presentes num NoAluno do aluno desejado. Recorre à função "**procura**" para obter os nós do aluno em questão;

"**imprime_saldo**" -> Apresenta os dados dos alunos que tenham saldo inferior a um determinado valor, imprimindo a lista, lista_saldo na íntegra;

"**ordena**" -> Ordena os alunos por ordem alfabética;

"**ordena_saldo**" -> Encontra o primeiro NoSaldo com menor valor de saldo que o fornecido pelo utilizador;

"**ordena_despesas**" -> Ordena as despesas de forma que as realizadas pelo mesmo aluno fiquem juntas;

"**procura**" -> Procura o aluno com o número pretendido percorrendo a lista de alunos com recurso a ponteiros, deixando estes a apontar para o aluno procurado;

"**procura_saldo**" -> Procura alunos com saldo inferior a um determinado valor percorrendo a lista de alunos com recurso a ponteiros, deixando estes a apontar para o aluno procurado;

"**escreve_ficheiro_alunos**" -> Guarda os dados dos alunos presentes em cada NoAluno da lista de alunos, linha a linha, no ficheiro "Alunos";

"**escreve_ficheiro_despesas**" -> Guarda os dados das despesas presentes em cada NoDespesa da lista de despesas, linha a linha, no ficheiro "Despesas";

3 - Estruturas:

- 3.1 – Dados:

Struct **Aluno**- estrutura que guarda a informação de cada aluno. É composta por:

- char Nome;
- char Data Nascimento;
- int Ano;
- char Turma;
- int Número;
- float Saldo

Struct **Despesas**- estrutura que guarda a informação de cada despesa. É composta por:

-int numero_aluno;
- float valor;
- char descrição;
Char data;

Struct **NoAluno-** Nó da lista de alunos que guarda a informação relativa a cada aluno e um ponteiro para o próximo nó. É composta por:

- struct Aluno aluno;
- struct NoAluno * prox;

Struct **NoSaldo-** Nó da lista saldo que tem um ponteiro para o aluno em questão e um ponteiro para o próximo nó**.** É composta por:

- struct NoAluno * no_aluno;
- struct NoSaldo * prox;

Struct **NoDespesas-** Nó da lista de despesas que guarda a informação relativa a cada despesa, um ponteiro para o aluno em questão e outro para o próximo nó. É composta por:

- struct Despesa despesa;
- struct NoAluno * no_aluno;
- struct NoDespesas * prox;

- 3.2 – Ficheiros:

Alunos:

1 Nome
2 Data Nascimento | Ano | Turma | Número Aluno | Saldo
3 \n
4 Nome
5 Data Nascimento | Ano | Turma | Número | Saldo
6 \n

Despesas:

1 Descrição
2 Número Aluno | Saldo | Data