**Manual do Programador**

O trabalho está organizado pelo main, e três headers: alunos, principal, auxiliar.

* Principais estruturas de dados:

- Durante a implementação das estruturas, foi usado uma estrutura principal, a estrutura do ***aluno*** que guarda as suas informações: nome, \*data de nascimento, \*\*turma, número de estudante, saldo, número de despesas e as \*\*\*despesas.

- Existe uma lista para os alunos que diz onde é o seu início. Com o início podemos mexer no aluno atual e no próximo.

*\*data de nascimento* é uma estrutura do tipo **data**, que só guarda o dia, mês e ano como int.

*\*\*turma* é uma estrutura que guarda o ano em int, e a sigla da turma que pode ser até 9 caracteres, números ou letras.

*\*\*\*despesas* é uma estrutura que guarda as despesas do aluno em forma de lista, ou seja, é permitido percorrer as despesas do aluno (funciona da mesma forma que a lista dos alunos).

* Principais funções:

- Esta secção está dívida em dois ficheiros headers diferentes: alunos, e principal.

1. No **header alunos** estão as funções: pede\_aluno, inserir\_aluno, eliminar\_aluno, procurar\_aluno, e carregar\_conta.

1.1. Começando pela função ***pede\_aluno***, não tem parâmetros e retorna o endereço do aluno que foi pedido.

Nesta função pede o nome com lenght máximo de 100 caracteres, data de nascimento, em que o utilizador insere a data com o formato dia/mês/ano, número de estudante tem 10 dígitos inteiros em que começa com o ano letivo atual e é seguido por outros 6 dígitos (e.g 2022XXXXXX), turma do aluno pedido no formato “ano sigla” os anos disponíveis são 10, 11 e 12, depois é verificado todos os inputs, e também confirma se o utilizador quer mesmo inserir o aluno, e se estiverem todos corretos é então devolvido o endereço para o aluno.

O aluno quando é inserido começa sem despesas e com o saldo a 0.

1.2. ***inserir\_aluno***, tem como parâmetros um ponteiro para a lista dos alunos, um ponteiro para o aluno que se pretende inserir, e o toggler, que informa se é para inserir na lista (toggler = 0), ou inserir na lista e no ficheiro (toggler = 1). A função devolve verdadeiro, 1, se conseguir fazer tudo, e devolve falso, 0, caso contrário.

1.3. ***eliminar\_aluno***, tem como parâmetros um ponteiro para a lista dos alunos, e o número do aluno que se quer remover da lista. Devolve 0 ou 1.

Nesta função é criada uma lista auxiliar, que vai ser percorrida até encontrar o aluno, se encontrar apaga o aluno e as suas despesas.

1.4. ***procurar\_aluno***, tem como parâmetros um ponteiro para a lista dos alunos, e o número do aluno que se quer encontrar. Devolve o endereço do aluno caso o encontre, e devolve NULL caso contrário.

1.5. ***carregar\_conta***, têm como parâmetros um ponteiro para a lista dos alunos, o número de estudante, e o montante que se pretende carregar.

Usa a função de **procurar\_aluno** e depois, se o aluno existir, carrega o saldo em cêntimos para que não haja erros em longo prazo.

2. No **header principal**, estão as funções: inicializa\_lista, inicializa\_despesa, criar\_despesa, reescreve\_ficheiro, inserir\_aluno\_ficheiro, funções de ordenar por saldo e alfabeticamente, e a função menu.

2.1. ***inicializa\_lista***, tem como parâmetros um ponteiro para a lista dos alunos, e devolve 1 se conseguir inicializar, e 0 caso contrário.

Nesta função primeiro cria a lista, apontando o seu início para NULL, e depois vê se o ficheiro tem algum aluno. Se o ficheiro não estiver criado, então cria logo o ficheiro.

No caso de ter alunos, lê o aluno no formato “nome: dd/mm/yyyy | ano | sigla | num\_estudante | saldo | num\_despesas“, depois percorre as depesas do aluno e lê com o formato “descrição: preço | dd/mm/yyyy”. Depois de lêr insere na lista (chama **inserir\_aluno**, com toggler = 0).

2.2. ***inicializa\_despesa***, tem como parâmetros um ponteiro para o aluno, e depois aloca memória para as despesas do aluno. Devolve 1 se conseguir, e 0 caso contrário.

2.3. ***cria\_despesa***, tem como parâmetros um ponteiro para a lista dos alunos, o número do aluno, a data da despesa, o preço da despesa, e a descrição. Devolve 1 se conseguir fazer a despesa, 0 se não houver espaço para a despesa, -1 se o aluno não tiver saldo suficiente.

Nesta função tenta procurar o aluno e se ele existir, remove-lhe saldo equivalente ao preço da despesa, e depois adiciona a despesa ao seu histórico por ordem de data (decrescente, mais recente para a mais antiga).

No final adiciona +1 ao número de despesas que o aluno tem.

2.4. ***reescreve\_ficheiro***, tem como parâmetro um ponteiro para a lista dos alunos.

Percorre a lista toda e vai reescrevendo o ficheiro pela ordem que está na lista. Devolve 1 se conseguir, e 0 caso contrário.

2.5. ***insere\_aluno\_ficheiro***, tem como parâmetros um ponteiro para o aluno, e devolve 1 se conseguir inserir ou 0 caso contrário.

Adiciona o novo aluno no final do ficheiro

2.6. ***funções de ordenar***, tem como parâmetros percorre a lista, e se uma condição se verificar troca a posição dos alunos na lista, usando uma variável temporária.

2.7 ***menu***, têm como parâmetro um ponteiro para a lista, dá hold a todos as opções.

* Como usar: …