Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamente

**Manual do Programador**

Princípios de Programação Procedimental

Feito por:

Filipe Rodrigues

Nº 2021228054

Joás Silva

Nº 2021226149

**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 de maneira parecida 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 possui parâmetros e retorna o endereço de memória do aluno que foi criado.

A função pede ao utilizado o nome do aluno (com lenght máximo de 100 caracteres), a data de nascimento (no formato dia/mês/ano), o número de estudante (int de 10 dígitos em que começa com o ano letivo atual e é seguido por outros 6 dígitos [e.g 2022XXXXXX]), a turma do aluno pedido no formato “ano sigla” (os anos disponíveis são 10, 11 e 12).

Após inseridos, todos os inputs são verificados, caso algum seja inválido, a função dá return a NULL.

É também pedida a confirmação se o utilizador quer mesmo inserir o aluno, e caso todos os inputs estejam 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 imprimir no ficheiro (toggler = 1).

Devolve 1 se conseguir inserir o aluno, devolve 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.

Utiliza a função **procurar\_aluno** para encontrar o aluno com o número de estudante inserido.

Nesta função é criada um ponteiro auxiliar, que vai percorrer a lista até encontrar o aluno, se o encontrar apaga-o junto com as suas despesas.

Devolve 1 se conseguir eliminar o aluno, devolve 0 caso contrário.

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, 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).

Devolve 1 se conseguir carregar o saldo do aluno, devolve 0 caso contrário.

2. No header **principal**, estão as funções: inicializa\_lista, inicializa\_despesa, criar\_despesa, reescreve\_ficheiro, inserir\_aluno\_ficheiro, ordena\_saldo, ordena\_alfabeticamente, e a função menu.

2.1. ***inicializa\_lista***, tem como parâmetros um ponteiro para a lista dos alunos.

Nesta função primeiro cria a lista, apontando o seu início para NULL, e depois vê se o ficheiro contém algum aluno. Se o ficheiro não existir cria-se um ficheiro “alunos.txt” vazio.

No caso de ter alunos, lê cada aluno no formato “nome: dd/mm/yyyy | ano | sigla | num\_estudante | saldo | num\_despesas“.

Após ler o aluno, percorre as suas despesas (o número de linhas de despesas do aluno está guardado na variável **num\_despesas**) e lê com o formato “descrição: preço | dd/mm/yyyy”.

Depois de ler o aluno, insere-o na lista (chama **inserir\_aluno**, com toggler = 0).

Devolve 1 se conseguir inicializar, devolve 0 caso contrário.

2.2. ***inicializa\_despesa***, tem como parâmetros um ponteiro para o aluno.

A função aloca memória para as despesas do aluno.

Devolve 1 se conseguir inicializar as despesas, 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.

Utiliza o **procurar\_aluno** para encontrar o aluno com o número de estudante inserido.

Caso o aluno exista, aloca espaço na lista de despesas do aluno e insere uma nova despesa na lista.

A nova despesa é ordenada em relação à data (as despesas são ordenadas das mais novas para as mais antigas).

No final adiciona 1 ao número de despesas que o aluno tem (num\_despesas).

Devolve 1 se conseguir fazer a despesa, 0 se não conseguir alocar espaço para a despesa, ou -1 se o aluno não tiver saldo suficiente.

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

Percorre toda a lista 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.

Adiciona o novo aluno ao final do ficheiro.

Devolve 1 se conseguir inserir, 0 caso contrário.

2.6. ***ordenar\_saldo***, tem como parâmetro um ponteiro para a lista de alunos.

Percorre a lista, ordenando-a por ordem decrescente de saldo.

Devolve 1 se conseguir ordenar, 0 caso contrário.

2.7. ***ordenar\_alfabeticamente***, tem como parâmetro um ponteiro para a lista de alunos.

Percorre a lista, ordenando-a alfabeticamente.

Devolve 1 se conseguir ordenar, 0 caso contrário.

2.8. ***menu***, têm como parâmetro um ponteiro para a lista de alunos.

Utiliza um switch para escolher o caso baseado no input do utilizador.

Devolve 1 se o utilizador ainda pretende utilizar o programa, 0 caso contrário.