Programação 2023/24

LEI, LEI-PL, LEI-CE

Aula Laboratorial 13

Bibliografia:

K. N. King. *C programming: A Modern Approach* (2nd Edition). W. W. Norton: capítulo 22.

Código de apoio para a aula: https://gist.github.com/FranciscoBPereira

Operações com Ficheiros

Exercícios Obrigatórios

1. O ficheiro *"dados.bin"* contém algumas estruturas do tipo pessoa. Considerando o ficheiro exemplo que é apresentado a seguir, que informação é escrita na consola ao ser executada a seguinte função? Pode assumir que não existe nenhum erro no acesso ao ficheiro.

```
typedef struct{
    char nome[100];
    int id;
}pessoa;
```

```
void f(){
    FILE *f;
    pessoa a;
    int x;

    f = fopen("dados.bin", "rb");
    if(f == NULL) return;
    fread(&a, sizeof(pessoa), 1, f);
    printf("%s\n", a.nome);
    fread(&a, sizeof(pessoa), 1, f);
    printf("%s\n", a.nome);
    fseek(f, 3 * sizeof(pessoa), SEEK_SET);
    x = ftell(f);
    fread(&a, sizeof(pessoa), 1, f);
    printf("%s\n", a.nome);
    fseek(f, 2*sizeof(pessoa), SEEK_CUR);
    fread(&a, sizeof(pessoa), 1, f);
    printf("%s\n", a.nome);
    fseek(f, -x, SEEK_END);
    fread(&a, sizeof(pessoa), 1, f);
    printf("%s\n", a.nome);
    fclose(f);
}
```

dados.bin

Ana
Bruno
Carolina
Diogo
Heitor
Luisa
Maria
Pedro

Programação 2023/24

LEI, LEI-PL, LEI-CE

2. Considere a seguinte definição:

```
typedef struct{
      char nome[200], morada[200];
      int conta, montante;
} cliente;
```

Um banco tem informação armazenada sobre os seus clientes num ficheiro binário contendo estruturas do tipo `cliente`. O ficheiro está **ordenado alfabeticamente** pelo nome do cliente. Pode assumir que não existem clientes diferentes com nomes iguais.

a) Escreva uma função em C que mostre no monitor os nomes e os números de conta de todos os clientes que fazem parte do ficheiro. O nome do ficheiro é passado como parâmetro da função.

b) Escreva uma função em C que mostre no monitor o número de bytes do ficheiro e o número de clientes que aí estão armazenados. O nome do ficheiro é passado como parâmetro da função.

c) Escreva uma função em C que corrija a morada de um dos clientes armazenado no ficheiro. A função recebe como parâmetros o nome do ficheiro, o nome do cliente e a sua nova morada. Devolve 1 se a correção for efetuada com sucesso, ou 0, caso contrário.

d) Escreva uma função em C que mostre no monitor os nomes e os números de conta de todos os clientes que fazem parte do ficheiro. A informação deve ser listada por ordem alfabética inversa. O nome do ficheiro é passado como parâmetro da função.

e) Escreva uma função em C que efetue uma transferência entre 2 clientes. A função tem o protótipo:

```
int transfere(char *nomeF, char *or, char *dest, int valor);
```

Transfere *valor* da conta do cliente *or* para o cliente *dest*. Ambos os clientes devem fazer parte do ficheiro *nomeF*. A função deve atualizar os montantes dos clientes, de acordo com o valor transferido. A transferência só é concretizada se existir saldo suficiente na conta do cliente *or*. Devolve 1 se tudo correu bem, ou 0, caso contrário.

f) Escreva uma função em C que elimine um cliente do ficheiro. A função recebe como parâmetros o nome do ficheiro e o nome do cliente a eliminar. Devolve 1 se tudo correu bem, ou 0, caso contrário.

g) Escreva uma função em C que adicione um novo cliente ao ficheiro. A função recebe como parâmetros o nome do ficheiro e a estrutura com a informação sobre o novo cliente (os campos

da estrutura já estão preenchidos). A função deve garantir que o ficheiro se mantém ordenado alfabeticamente depois da inserção do novo cliente. Devolve 1 se tudo correu bem, ou 0, caso contrário.

h) Considere que o banco dividiu por dois ficheiros a informação sobre os seus clientes. Em cada um deles as estruturas estão armazenadas por ordem alfabética do nome do cliente (recorde que não existem clientes diferentes com nomes iguais). O banco pretende reunir as estruturas num ficheiro único. Escreva uma função em C para realizar esta tarefa. Os nomes dos dois ficheiros originais e do novo ficheiro são passados como parâmetros. No novo ficheiro, as estruturas devem manter a ordem alfabética. Podem existir duplicações da informação de um cliente (ou seja, o mesmo cliente tem uma estrutura em cada um dos ficheiros originais). Neste caso, só uma estrutura deve ser escrita no novo ficheiro e o valor do montante deve ser a soma dos dois montantes existentes em cada uma das cópias. Pode assumir que nestes casos a morada e o número de conta são iguais.

3. Considere a seguinte definição:

```
struct cliente{
      int id;           // id numérico do cliente
      char cc[10];      // Número do cartão de cidadão
      float valor;      // Valor gasto em compras desde o início do mês
      float v_divida;   // Valor em divida desde o início do mês
      char cCert;       // Cliente certificado (S/N)
};
```

Uma pequena mercearia de bairro possui um ficheiro binário onde armazena informação sobre os seus clientes. A informação de cada cliente está armazenada em estruturas do tipo *struct cliente* e a organização do ficheiro é a seguinte: no início estão armazenados 2 inteiros A e B, o primeiro (A) indicando qual o número de clientes armazenados no ficheiro e o segundo (B) indicando quantos destes são clientes certificados. A seguir estão armazenadas A estruturas do tipo *struct cliente*. O ficheiro não está ordenado por nenhum critério em particular. Nesta altura ainda não foram contabilizados os clientes certificados. Todos os campos *cCert* das estruturas *struct cliente* têm o valor 'X' e o inteiro B tem o valor 0.

a) Escreva uma função em C que calcule as médias do valor gasto em compras desde o início do mês e do valor em dívida desde o início do mês. A função recebe o nome do ficheiro como parâmetro e devolve os 2 valores contabilizados.

b) Escreva uma função em C que atualize a informação que se encontra no ficheiro binário. Devem ser efetuadas 3 atualizações:

i. Devido à lei de proteção de dados é ilegal guardar o número de cartão de cidadão de clientes. Desta forma, todos os campos *cc* dos clientes devem passar a armazenar a string "NA".
ii. Um cliente é considerado certificado se tiver um valor em dívida inferior a 10% do valor gasto em compras desde o início do mês. Com base nesta regra, atualize o campo *cCert* de todos os clientes armazenados no ficheiro (S/N).
iii. Atualize o valor B que se encontra no início do ficheiro, com a indicação de quantos clientes estão certificados.

A função recebe o nome do ficheiro como parâmetro.

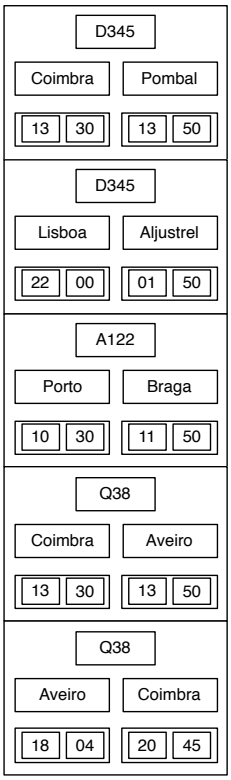
4. Considere as seguintes definições:

```
struct hora{
    int h, m;
};

typedef struct{
    char id[15];
    char in[50], out[50];  // locais de entrada/saída
    struct hora inH, outH; // horas de entrada/saída
}util;

typedef struct{
    char id[15];
    int valorTotal;
}resumo;
```

As utilizações do sistema Via Verde durante um determinado período estão armazenas num ficheiro binário com estruturas do tipo *util*.
Cada estrutura refere-se a uma utilização completa do sistema contendo a identificação do utilizador, os locais de entrada e saída e as respetivas horas de entrada e saída. Ao responder a esta questão pode assumir que nenhuma utilização individual ultrapassa as 12 horas e que os locais de entrada e saída são constituídos apenas por uma palavra. No ficheiro exemplificado ao lado existem 5 utilizações de 3 utilizadores diferentes.



a) Escreva uma função em C que contabilize o número de utilizações da Via Verde que foram efetuadas por um determinado utilizador. A função recebe como parâmetros o nome do ficheiro e a identificação do utilizador. Devolve o número de utilizações contabilizado.

b) Escreva uma função em C que elimine do ficheiro binário todas as utilizações inferiores a uma hora. A função recebe como parâmetro o nome do ficheiro. Deve atualizar o ficheiro da forma pretendida e devolver o número de estruturas eliminadas.

c) Um ficheiro de texto armazena os valores a pagar entre todos os pares entrada/saída que podem ser usados na autoestrada. Em cada linha deste ficheiro surge a identificação de 2 locais seguidos pelo valor inteiro a pagar, como se pode ver no exemplo em baixo. Todos os veículos pagam o mesmo e a direção seguida é indiferente (Aveiro-Coimbra tem o mesmo custo de Coimbra-Aveiro).

```
Lisboa   Coimbra   12
Aveiro   Coimbra   5
...
```

Escreva uma função em C que crie um vetor dinâmico do tipo *resumo* com informação sobre o valor a pagar por cada utilizador no período registado no ficheiro binário. Cada estrutura pertencente ao vetor deve armazenar os dados de um utilizador, sendo o valor total a pagar contabilizado pela função com ajuda da informação existente no ficheiro de texto. Considerando o exemplo em cima, deveria ser criado um vetor dinâmico com 3 estruturas. O cabeçalho da função é o seguinte:

```
resumo* criaVetor(char* nomeBin, char* nomeTXT, int* total);
```

Devolve o endereço inicial do vetor dinâmico criado dentro da função (NULL caso exista algum erro) e recebe como parâmetros os nomes dos ficheiros e o endereço de uma variável inteira onde deve ser colocada a dimensão do vetor.

Exercícios Complementares

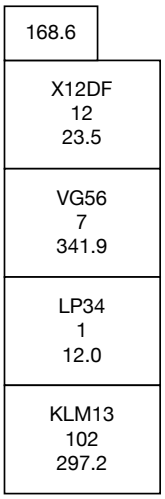
5. Considere a seguinte definição:

```
struct artigo{
    char ref[10];
    int qtd;
    float preco;
};
```

Um armazém guarda num ficheiro binário informação sobre os produtos que disponibiliza, incluindo as quantidades em stock e o preço unitário.
O ficheiro tem a seguinte organização:

i. No início surge um valor real de precisão simples (*float*) contendo o preço médio dos artigos armazenados no ficheiro.
ii. A seguir surgem várias estruturas do tipo *struct artigo*, uma para cada um dos produtos existentes.

Ao lado pode consultar um exemplo de um ficheiro com 4 produtos.



Escreva uma função em C que efetue as seguintes operações:

i. Atualize o preço de todos os produtos do ficheiro binário. Todos os preços devem ser aumentados 10%. O preço médio que surge no início do ficheiro também deve ser atualizado.
ii. Crie um ficheiro de texto, para onde deve copiar informação sobre todos os produtos que tenham um stock inferior a um determinado limite. No ficheiro de texto que for criado, deve surgir em cada linha a referência e quantidade em stock dos produtos relevantes (informação separada por um ou mais espaços em branco).

Considerando o ficheiro exemplificado, e assumindo que o valor limite é 10, deveria ser criado um ficheiro de texto com a seguinte organização:

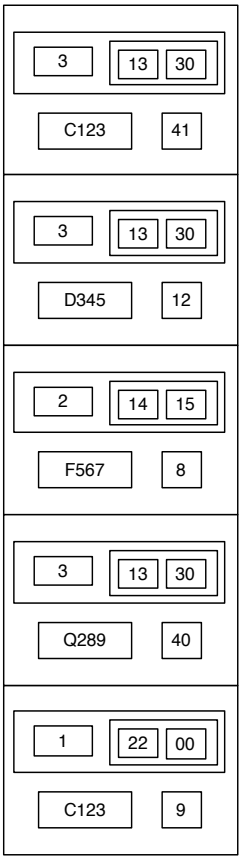
```
VG56  7
LP34  1
```

A função recebe como parâmetros os nomes dos ficheiros e o valor limite para a quantidade em stock a considerar. Devolve o número de produtos colocados no ficheiro de texto. Em caso de erro, devolve -1.

6. Um cinema guarda informação sobre todos os bilhetes vendidos ao longo de um dia num ficheiro binário. O ficheiro é constituído por estruturas do tipo *struct bilhete.* Cada uma destas estruturas guarda informação sobre um bilhete vendido: qual a sessão para o qual foi vendido (sala e hora), identificação do cliente e lugar.

```
struct sessao{
    int sala;
    struct{int h, m;} hora;
};

struct bilhete{
    struct sessao s;
    char id[10];
    int lugar;
};
```



No ficheiro exemplificado ao lado existem 5 bilhetes vendidos distribuídos por 3 sessões diferentes. Nos ficheiros a processar nesta questão não se sabe, à partida, quantas sessões existem ou quantos bilhetes foram vendidos para cada uma delas. Cada sessão é identificada de forma única pela sala e horário em que decorre.

a) Escreva uma função em C que transfira para um ficheiro de texto a identificação de todos os clientes que compraram bilhete para uma determinada sessão (1 id por linha). A função recebe como parâmetros o nome do ficheiro binário onde estão os bilhetes vendidos nesse dia, o nome do ficheiro de texto a criar e uma estrutura do tipo *struct sessao* que identifica a sessão a considerar. Devolve o número de clientes escritos no ficheiro, ou -1, em caso de erro.

b) Escreva uma função em C que crie um vetor dinâmico do tipo *struct sessao* com informação sobre todas as sessões existentes nesse dia. Cada estrutura pertencente ao vetor deve armazenar os dados de uma das sessões existentes. Considerando o exemplo em cima, deveria ser criado um vetor dinâmico com 3 estruturas. O cabeçalho da função é o seguinte:

```
struct sessao* criaVetor(char* nome, int* total);
```

Devolve o endereço inicial do vetor dinâmico criado dentro da função (NULL caso exista algum erro) e recebe como parâmetros o nome do ficheiro binário e o endereço de uma variável inteira onde deve ser colocada a dimensão do vetor.

c) Escreva uma função em C que devolva o número de pessoas que assistiram à sessão para a qual foram vendidos mais bilhetes. A função recebe o nome do ficheiro binário como parâmetro.

Nota: Caso o ajude a resolver esta questão, pode usar a função implementada na alínea anterior. Mesmo que não a tenha implementado, conhece o protótipo e pode usá-la assumindo que funciona corretamente.