Programação - Exame da época normal

Engª Informática; Engª Informática – Pós-laboral; Engª Informática - Curso Europeu

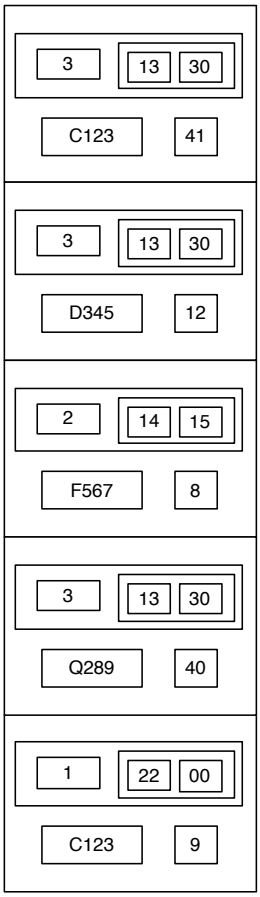
Duração: 2h30m 28/06/2017

Atenção: É obrigatório apresentar uma estratégia genérica para cada um dos exercícios.

1. 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) Desenvolva 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 todos 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 no caso de algum erro ter ocorrido.

b) Desenvolva 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) Desenvolva 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.

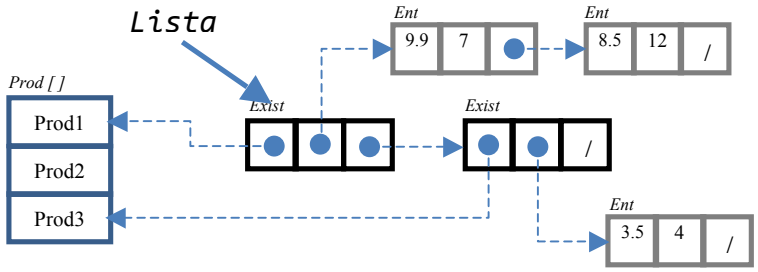
2. Considere as seguintes definições que permitem criar uma estrutura dinâmica para armazenar produtos de um supermercado:

```
typedef struct produto Prod, *pProd;
typedef struct entrada Ent, *pEnt;
typedef struct existencia Exist, *pExist;

struct produto {         // representa um produto
  int id;                // identificador único do produto
  char designacao[100];
  char familia[30];      // fruta, detergente, lacticínios, ...
};

struct entrada {           // representa uma entrada de stock
  float qt, preco_compra;  // qtd. que deu entrada e preço unitário
  pEnt prox;
};

struct existencia { // representa o conjunto de abastecimentos de um produto
  pProd p;          // produto a que se refere esta estrutura
  pEnt entradas;    //lista com as diversas entradas deste produto
 pExist prox;
};
```



A estrutura dinâmica é constituída por um vetor dinâmico de *Prod* onde se encontram armazenados todos os produtos. Existe igualmente uma lista ligada constituída por nós do tipo *Exist* com as existências dos produtos. Cada nó da lista principal referencia um produto específico (através do ponteiro *p*) e tem pendurada uma lista de entradas em stock desse produto. Cada entrada em stock é caracterizada por uma quantidade (número de unidades adquiridas) e pelo preço unitário.

a) Desenvolva uma função em C que calcule o preço a que um determinado produto deve ser vendido. O preço deve ser calculado com base na média ponderada de todos os preços de compra desse produto e respetivas quantidades. Por exemplo, se foram compradas 4 unidades de um produto a 10€, 5 unidades a 12€ e 1 unidade a 15€, o preço de venda deve ser 11,50€ = (4*10+5*12+1*15)/(4+5+1). A função recebe como parâmetros o ponteiro para a lista ligada de existências de todos os produtos e o *id* do produto a considerar. Devolve o preço calculado ou -1 se o produto não existir.

b) Desenvolva uma função em C que retire da estrutura dinâmica algumas unidades de um determinado produto. A função recebe como parâmetros o ponteiro para a lista ligada de existências de todos os produtos, o *id* do produto a considerar e a quantidade a retirar de stock. Devolve o ponteiro para o início da lista depois da alteração. Pode assumir que a quantidade a retirar é sempre igual ou inferior ao somatório das unidades existentes. Tenha em consideração que poderá ter que retirar várias entradas do produto para atingir a quantidade pretendida – por exemplo, se existir uma entrada com 4 unidades e outra com 6 unidades, para atingir a quantidade 8 será necessário eliminar a primeira entrada e retirar mais 4 unidades da segunda. As entradas que ficam com quantidade 0 devem ser eliminadas. Se este processo eliminar todas as entradas do produto, o nó respetivo da lista de existências também deve ser eliminado.