Programação - Exame da época normal

Eng.ª Informática; Eng.ª Informática – Pós-laboral; Eng.ª Informática - Curso Europeu

Duração: 2h30m 02/07/2018

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

---

1. Um ficheiro binário armazena informação sobre tarefas realizadas por uma empresa de limpeza. Cada tarefa possui um identificador único (inteiro positivo), uma designação e um valor correspondente ao preço para a realização da tarefa em causa. No ficheiro binário, estas tarefas são armazenadas usando a estrutura *Tarefa*, conforme a definição seguinte:

```
typedef struct tarefa {
      int id;
      char designacao[100];
      float preco;
} Tarefa;
```

a) Desenvolva uma função em C que apresente na consola a informação completa das tarefas cujo identificador esteja compreendido entre 2 limites. Deve devolver o número total de tarefas armazenadas no ficheiro e o preço da tarefa de menor custo que foi apresentada na consola. A função recebe como parâmetros o nome do ficheiro binário onde estão armazenadas as tarefas e os limites a considerar para o identificador.

b) A empresa de limpeza possui um ficheiro de texto onde estão discriminadas as identificações das tarefas que os seus empregados irão realizar durante um determinado dia. Este ficheiro possui uma linha para cada funcionário com um número variável de tarefas. Cada linha segue o formato exemplificado a seguir e é finalizada com o identificador 0:

```
Manuel Silva, 23, 34, 12, 0
Olga Pereira, 7, 23, 0
```

Desenvolva uma função em C que apresente na consola uma listagem com as designações das tarefas a realizar por cada funcionário, ou seja, algo do género:

Manuel Silva:
- [23] Limpar as escadas
- [34] Fazer as camas
- [12] Lavar os vidros

Olga Pereira:
- [7] Aspirar os tapetes
- [23] Limpar as escadas

A função recebe os nomes dos ficheiros como parâmetros e devolve o somatório dos preços de todas as tarefas previstas para esse dia.

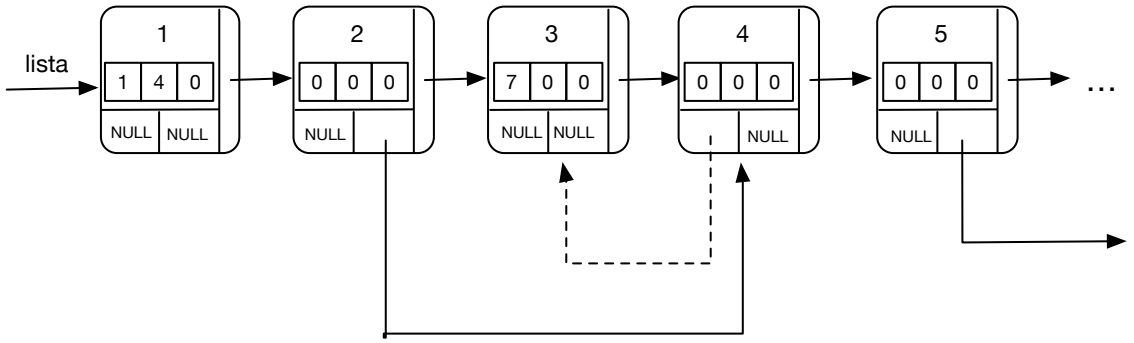
2. Escreva uma **função recursiva** em C que verifique se um determinado valor se encontra num vetor ordenado de inteiros. A função tem o seguinte protótipo:

```
int procuraOrd(int *tab, int tam, int valor);
```

Recebe como parâmetros o vetor, a sua dimensão e o valor a pesquisar. Devolve 1 se o valor existir, ou 0, caso contrário.

3. Uma estrutura dinâmica armazena um jogo de tabuleiro do tipo jogo da glória. Neste jogo, vários jogadores (3, no máximo) lançam alternadamente um dado para ir avançando nas casas de um tabuleiro. Ganha quem atingir primeiro a última posição. Em algumas casas existem escadas: um jogador que atinja uma escada durante uma jogada, avança para uma casa mais à frente. Em outras posições existem cobras que fazem um jogador recuar algumas casas. Nenhuma posição tem simultaneamente uma escada e uma cobra e não existem cobras nem escadas nas primeira e última posições do tabuleiro. A estrutura seguinte define uma posição, i.e., uma casa do tabuleiro, e permite criar a estrutura dinâmica completa para jogar o jogo:

```
typedef struct posicao no, *pno;
struct posicao{
    int index;    // Índice da posição do tabuleiro (a 1ª tem índice 1)
    int joga[3];  // Ids dos jogadores que estão nesta posição
    pno cobra, escada;  // cobras e escadas existentes nesta posição
    pno prox;
};
```



A estrutura dinâmica exemplificada em cima mostra as primeiras 5 posições de um tabuleiro. São casas sequenciais identificadas pelo seu índice e, em cada casa, existem 2 ponteiros que assinalam a existência de eventuais cobras ou escadas. Se algum destes ponteiros estiver a NULL, não existe ligação deste tipo. Caso contrário apontam para a posição para onde a cobra ou escada levam. No exemplo da figura, a posição 2 tem uma escada para a posição 4. Por sua vez a posição 4 tem uma cobra para a posição 3. Em cada uma das posições do tabuleiro, a tabela *joga* assinala que jogadores se encontram nesse local. Cada jogador tem um identificador inteiro positivo para esse efeito. No exemplo, os jogadores 1 e 4 estão na posição 1 e o jogador 7 está na posição 3. A tabela *joga* serve apenas para identificar quais os jogadores que se encontram numa determinada posição, pelo que a ordem pela qual os identificadores estão armazenados não é relevante ({1, 4, 0} é equivalente a {4, 0, 1}).

a) Desenvolva uma função em C efetue uma jogada. A função tem o seguinte protótipo:

```
int jogada(pno lista, int totPos, int idJog, int dado);
```

Recebe um ponteiro para o início da estrutura dinâmica, o número de casas do tabuleiro (o número de nós da lista), a identificação do jogador que está a fazer a jogada e o valor que saiu no dado. Deve atualizar a estrutura dinâmica movendo o jogador para a nova posição. Devolve a casa em que ele ficar depois da jogada. Se o jogador indicado não existir, a função devolve 0.
Caso a jogada o faça ultrapassar o limite do tabuleiro, deve ficar colocado na última casa desse mesmo tabuleiro. Por exemplo: se o jogador 4 tiver 1 no dado, deve sair da posição 1 e terminar a jogada na posição 3.

b) Desenvolva uma função em C que faça uma cópia de uma estrutura dinâmica deste tipo. A função recebe um ponteiro para o início da estrutura dinâmica e o número de casas do tabuleiro. Devolve um ponteiro para uma nova estrutura dinâmica que seja uma cópia da recebida por parâmetro. A estrutura dinâmica original não deve ser modificada.