Programação - Exame da época especial

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

Duração: 2h30m 07/09/2016

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

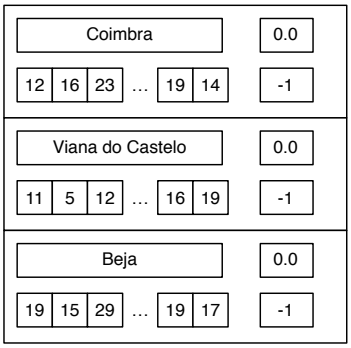
---

1. Um ficheiro de texto armazena uma matriz quadrada de valores inteiros, de acordo com o seguinte formato: na primeira linha surge a dimensão da matriz e nas linhas seguintes surgem os valores armazenados. No início de cada uma destas linhas surge a indicação "Linha X:" em que X é o índice numérico da linha, seguido pelos valores inteiros separados por um ou mais espaços em branco. O exemplo seguinte ilustra uma matriz 4×4:

```
DIM: 4
Linha 0: 12  4  6  3
Linha 1: 3  7  3  5
Linha 2: -56  31  31  12
Linha 3: 34  -33  2  4
```

Desenvolva uma função em C que receba como argumento o nome de um ficheiro de texto contendo uma matriz neste formato e que verifique se existem 2 inteiros iguais, imediatamente um a seguir ao outro, na mesma linha. A função devolve 1 se existirem 2 inteiros consecutivos iguais numa mesma linha, ou 0 caso contrário. No exemplo em cima, a função deveria devolver 1.

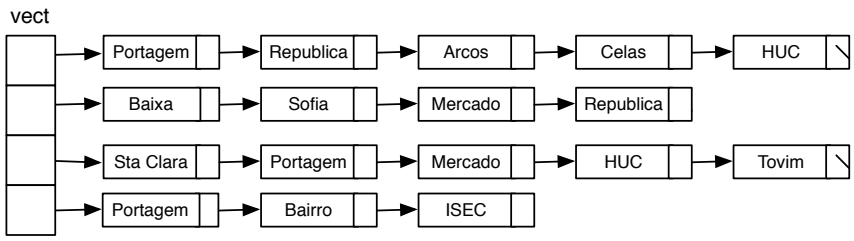
2. Um ficheiro binário contém informação sobre a temperatura mais elevada que se verificou em cada um dos 12 meses de um ano, para diversas cidades portuguesas. A informação relativa a cada cidade está guardada numa estrutura do tipo *struct t.*




```
struct t {
    char cidade[50];
    int temp[12];
    float media;
    int conta;
};
```

No ficheiro todas as estruturas têm os campos *cidade* e *temp* preenchidos, respetivamente com o nome da cidade e as 12 temperaturas registadas ao longo do ano. No entanto, falta preencher em cada uma das estruturas o campo *media* com o valor médio da temperatura que se verificou nessa cidade e o campo *conta* com a contabilização do número de meses em que a temperatura foi inferior à média nessa cidade.
Desenvolva uma função em C que receba o nome do ficheiro binário como argumento e atualize a informação armazenada, completando os 2 valores em falta em todas as estruturas.

3. Os percursos seguidos pelos autocarros dos SMTUC estão armazenados numa estrutura dinâmica como a da figura:




```
typedef struct paragem
no, *pno;

struct paragem{

   char nome[50];

   pno prox;

};
```

Existe uma lista ligada constituída por nós do tipo *struct paragem* para cada percurso e os vários percursos são acessíveis a partir de ponteiros do tipo *pno* guardados num vetor de ponteiros dinâmico. Na figura pode ver-se um exemplo com 4 percursos, sendo que o primeiro se inicia na Portagem, passa pela Republica, Arcos e Celas, terminando nos HUC. Os percursos não devem ser considerados circulares, i.e., iniciam-se no primeiro local e terminam no último, não voltando ao início.

a) Desenvolva uma função em C que verifique quantos percursos permitem fazer a ligação entre dois locais. Ao escrever a função considere que a ligação deve ser direta, ou seja, não implica mudança de autocarro. Por exemplo, se o ponto de partida for a Portagem e o destino for os HUC, dois dos percursos indicados em cima permitem fazer a ligação. A função recebe como argumentos um ponteiro para o início do vetor dinâmico, o número de percursos existentes e 2 *strings* indicando o ponto de partida e o ponto de chegada. Devolve o número de percursos que permitem fazer a ligação.

b) Algumas paragens estão em obras e não podem ser utilizadas. Desenvolva uma função em C que retire das listas ligadas os nós relativos a estas paragens e liberte o espaço ocupado por eles. Além disso, a função deve verificar se algum dos percursos fica vazio depois de eliminadas todas as paragens indicadas. Caso isso aconteça, esse percurso deve desaparecer da estrutura dinâmica, realocando o vetor de ponteiros e atualizando a variável inteira que armazena o número de elementos do vetor. Considerando o exemplo da figura, se as paragens *Portagem, Bairro e ISEC* estiverem em obras, o último percurso ficará vazio e o vetor dinâmico deverá passar a ter apenas 3 posições para referenciar os restantes percursos que se mantêm. A função tem o seguinte cabeçalho:

```
pno* atualiza(pno v[], int* tam, char* paragens[], int num_paragens);
```

Recebe como argumentos um ponteiro para o início do vetor dinâmico, o endereço da variável inteira contendo o número de percursos original, um ponteiro para um vetor de *char** referenciando o nome das paragens em obras e um inteiro indicando quantas paragens em obras existem. Devolve um ponteiro para o início do vetor dinâmico atualizado. O número de percursos que se mantêm no vetor deve ser atualizado através do segundo argumento.