Programação - Exame da época normal

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

Duração: 2h30m 22/06/2015

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

---

1. Pretende-se automatizar o processo de avaliação dos trabalhos da unidade curricular de Programação, nomeadamente na contabilização dos alunos que realizaram o trabalho prático.

Considere que existe um ficheiro binário no qual estão armazenadas estruturas do tipo *struct aluno* que representam cada um dos alunos inscritos na unidade curricular. Os alunos entregaram os seus trabalhos através de um ficheiro zip, cujo nome respeita o solicitado no enunciado, ou seja:

```
struct aluno {
   int numero;
   char nome[100];
};
```

Prog_P<*numturma*>_<*numaluno1*>_<*numaluno2*>.zip

Os nomes destes ficheiros foram colocados num ficheiro de texto (um nome por linha).

Exemplo:

```
Prog_P10_21210123_12120321.zip
Prog_P6_22110213_11221302.zip
```

Desenvolva uma função em C que receba os nomes dos 2 ficheiros e que gere uma listagem dos trabalhos na consola com o seguinte formato exemplificativo:

```
Grupo 1 [P10]:
      21210123 - Felisberto Pancrácio
      12120321 - Zeferino Anacleto

Grupo 2 [P6]:
      22110213 - Justino Ambrósio
      11221302 - Esmeraldo Rúbi
```

A numeração dos grupos é feita automaticamente respeitando a ordem pela qual aparece a informação no ficheiro de texto. Pode assumir que não existem alunos a fazer o trabalho individualmente e que os elementos de cada grupo pertencem à mesma turma.

Nota: Resoluções que façam leituras individuais dos caracteres armazenados no ficheiro de texto serão penalizadas. Sugere-se que utilize funções que permitam separar os diversos campos necessários à geração da listagem solicitada como, por exemplo, a função fscanf().

2. Desenvolva uma **função recursiva em C** que verifique se todos os nós de uma árvore binária do tipo *struct no* armazenam o mesmo valor. A função recebe um ponteiro para a raiz e devolve 1 se todos os valores armazenados forem iguais, ou 0, caso contrário.

```
struct no{
      int val;
      struct no *esq, *dir;
};
```

Ao responder a esta questão pode assumir que a árvore recebida é completa, ou seja, todos os nós que não são folhas têm 2 filhos.

3. Considere as seguintes definições:

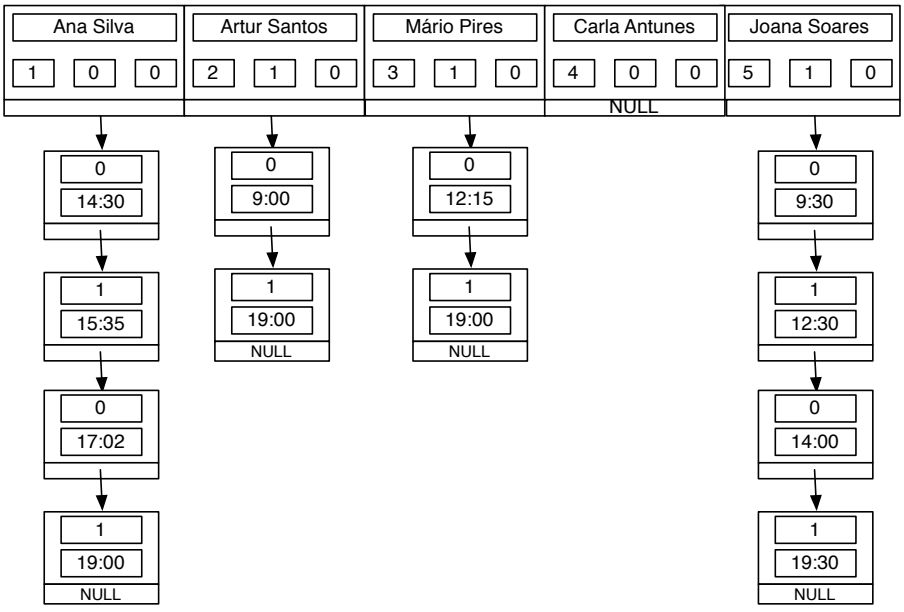
```
typedef struct tipoA cliente, *pcliente;
typedef struct tipoB acesso, *pacesso;

struct tipoA{
      char nome[100];          // Nome
      int id;                  // Identificador único
      int debito_direto;       // Débito direto (0:Não; 1:Sim)
      int total_m;             // Total de minutos no parque
      pacesso lista;           // Ponteiro para a lista de acessos
};

struct tipoB{
      int in_out;                    // (0:Entrada; 1:Saída)
      struct {int h, m;} hora;       // Hora do acesso
      pacesso prox;
};
```

Um parque de estacionamento armazena numa estrutura dinâmica informação sobre a utilização dos seus serviços por parte dos clientes registados. **Esta informação diz respeito a um único dia.**

Existe um vetor dinâmico de estruturas do tipo *cliente* contendo informação sobre os clientes habituais do parque. De cada estrutura deste vetor sai uma lista ligada constituída por elementos do tipo *acesso* contendo informação sobre os acessos desse cliente ao longo do dia. Cada nó de acesso regista a hora em que se processou a entrada ou saída do parque.



No exemplo da figura pode ver-se que a cliente Ana Silva (com identificador 1 e sem débito direto) entrou no parque às 14.30, saiu às 15.35, voltou a entrar às 17.02 e saiu às 19.00. Ao responder a esta questão pode assumir que as listas de acesso estão corretas: têm pares consecutivos de nós em que o primeiro corresponde a uma entrada e o seguinte à correspondente saída, sendo a hora registada no segundo superior à do anterior.

a) Desenvolva uma função em C que, para cada um dos clientes armazenados no vetor, calcule o número total de minutos que passou no parque durante o dia e preencha o campo *total_m* da respetiva estrutura. Pode assumir que no início todos os campos *total_m* têm o valor 0. A função recebe como argumentos um ponteiro para o início do vetor de clientes e o número de estruturas que este contém. No exemplo da figura, o valor a colocar no campo *total_m* da Ana Silva é 183.

b) Desenvolva uma função em C que elimine da estrutura dinâmica a informação dos clientes que fazem o pagamento por débito direto. Todo o espaço ocupado com informação destes clientes deve ser libertado. A função tem o seguinte cabeçalho:

```
pcliente elimina_direto(pcliente v, int *total);
```

Recebe um ponteiro para o início do vetor e um ponteiro para um inteiro que indica o número de estruturas que o vetor contém. A função devolve um ponteiro para o início do vetor depois de eliminar a informação. Deve também atualizar o número de estruturas que ficaram no vetor.