Programação – Exame de Recurso

03 de julho de 2024 – Duração: 120 minutos

LEI, LEI-PL, LEI-CE

---

1. Considere as seguintes definições:

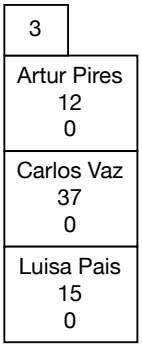
```
typedef struct a autor;

struct a{
    char nome[100];     // Nome do autor
    int idA;            // Identificador numérico do autor
    int nLivros;        // Número de títulos no catálogo da livraria
};
```

Uma livraria armazena em ficheiros informação sobre os autores que comercializa e os livros que tem disponíveis no seu catálogo. Um ficheiro binário guarda informação sobre os autores. A organização deste ficheiro é a seguinte: no início encontra-se um inteiro X indicando quantos autores diferentes comercializa. Seguem-se X estruturas do tipo *autor*, cada uma delas com dados de um dos autores comercializados. O campo *nLivros* desta estrutura especifica quantos títulos deste autor estão no catálogo da livraria. Nesta altura, a contabilização ainda não foi efetuada e todos os autores têm este campo com o valor 0.
Num ficheiro de texto estão armazenados os livros comercializados. Cada linha contém informação sobre um livro, no formato seguinte:

```
Título do livro # Id numérico do Autor # Preço (valor real)
```

Pode assumir que o ficheiro de texto só contém livros de autores que se encontram numa das estruturas do ficheiro binário. A informação nos ficheiros não está ordenada por nenhum critério em particular. A seguir pode consultar um exemplo em que os ficheiros armazenam 3 autores e 5 livros.

A Casa dos Gatos # 12 # 20.5
O Jardim # 37 # 12.5
Coimbra sem Sol # 12 # 15.4
A Praia # 12 # 12.0
Natal em Novembro # 12 # 25.9



Escreva uma função em C que efetue as seguintes operações:

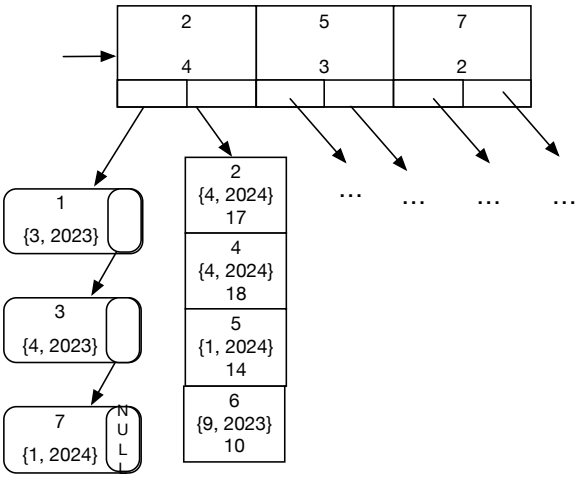
i) Atualize o campo *nLivros* em todas as estruturas do ficheiro binário, tendo em consideração os livros armazenados no ficheiro de texto. Por exemplo, considerando os ficheiros exemplificados, o autor Artur Pires deve passar a ter 4 livros registados e o autor Carlos Vaz deve passar a ter 1 livro registado.
ii) Escreva na consola o preço médio dos livros registados para cada um dos autores pertencentes ao catálogo da livraria.

A função recebe como parâmetros os nomes dos ficheiros. Devolve 1 se tudo correr bem, ou 0, no caso de existir algum problema no acesso aos ficheiros.

[Cotação: 30%]

2. Considere as seguintes definições:

```
typedef struct {int mes, ano;} data;
typedef struct emCurso no, *pno;

struct emCurso{
    int id;
    data inicio;
    pno prox;
};

struct completo{
    int id;
    data final;
    int duracao;  //duração em meses
};

struct gestor{
    int idG;
    pno lista;
    struct completo *v;
    int total;
};
```



Um gabinete de gestão de projetos armazena informação numa estrutura dinâmica. Um vetor dinâmico de estruturas do tipo *struct gestor* armazena informação sobre os gestores (1 estrutura por gestor). Um gestor é identificado pelo seu id numérico único e o vetor está ordenado por este valor. Cada gestor mantém um registo dos seus projetos em curso e projetos terminados:

i. Os projetos em curso estão armazenados numa lista ligada simples constituída por nós do tipo *no*, sendo caracterizados pelo seu identificador numérico único e pela data de início do projeto (mês e ano). O campo *lista* da estrutura *struct gestor* permite aceder a esta lista.
ii. Os projetos terminados estão armazenados num vetor dinâmico de estruturas do tipo *struct completo*. Cada projeto completo é identificado pelo seu identificador numérico único, pela data em que terminou (mês, ano) e pela duração em meses. Na estrutura *struct gestor*, o campo *v* permite aceder ao vetor e o campo *total* indica o número de projetos concluídos.

Tanto as listas de projetos em curso, como os vetores de projetos terminados, estão **ordenados pelo identificador de projeto**. Os identificadores são únicos para todo o gabinete de projetos, ou seja, não há repetição de identificador. Na figura em cima exemplifica-se uma situação com 3 gestores. O primeiro gestor, com id2, tem 3 projetos em curso e 4 projetos terminados.

2a) Escreva uma função em C que imprima na consola o identificador numérico dos gestores que tenham mais projetos em curso do que terminados. A função recebe, como parâmetros, o endereço do início do vetor de gestores e a sua dimensão.

[Cotação: 20%]

2b) Escreva uma função em C que finalize um projeto que esteja em curso, ou seja, que o transfira para o respetivo vetor de projetos terminados. A função tem o seguinte protótipo:

```
int finaliza(struct gestor *a, int tam, int id, data atual);
```

A função recebe, como parâmetros, o endereço do início do vetor de gestores, a sua dimensão, o identificador do projeto a transferir e a data atual. Deve retirar o projeto da lista ligada em que se encontra (libertando o espaço) e atualizar o vetor de projetos finalizados do gestor ao qual o projeto está associado. Devolve 1 se tudo correr bem, 0 se o projeto não existir em nenhuma lista de projetos em curso, ou -1 se suceder algum erro de gestão dinâmica.

[Cotação: 40%]

Programação – Exame de Recurso

3 de julho de 2024 – Duração: 120 minutos

LEI, LEI-PL, LEI-CE

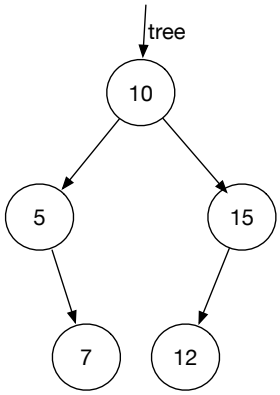
Nome: _____ **Número:** _____

4. Funções Recursivas

Considere a definição do tipo de dados seguinte, usado para criar uma árvore binária constituída por elementos do tipo *no*:

```
typedef struct elemento no, *pno;
struct elemento{
    int valor;
    pno esq, dir;
};
```

A função *f( )* recebe como parâmetros um ponteiro para a raiz de uma árvore binária ordenada e um valor inteiro.




```
void f(pno tree, int val){
    if(tree == NULL || val <= 0)
        return;
    else{
        if(val%2 == 0)
            f(tree->esq, val/10);
        else
            f(tree->dir, val/10);
        printf("%d\t", tree->valor);
    }
}
```

Considerando que o ponteiro *tree* aponta para a raiz da árvore ilustrada em cima, qual é o output na consola quando é efetuada a seguinte chamada da função *f( )*?

```
f(tree, 132);
```

Resposta:

a) ☐ 10 5 7

b) ☐ 10 15 12

c) ☐ 7 5 10

d) ☐ 10 15

e) ☐ 12 15 10

f) ☐ 15 10

[Cotação: 10%] (uma resposta errada não desconta)