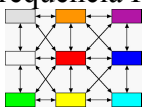
Estruturas de Dados
2016/17
Frequência F2



Universidade da Beira Interior
Departamento de Informática

1. Considere a seguinte estrutura de dados binária:

```
typedef struct NODOAB{
        int codigo;       //chave de ordenação na árvore
        char nome[80];
        struct NODOAB *fe, *fd;
}NodoAb;
```

1a) (2 valores) Mediante a estrutura apresentada, esquematize a organização da informação resultante do armazenamento sequencial dos seguintes itens:

- Código 2, José Andrade;
- Código 3, André Almeida;
- Código 4, André Soares;
- Código 1, Paula Correia;
- Código 5, Vitor Santos;
- Código 7, Hugo Filipe;

1b) (2 valores) Como verificou, a estrutura binária não fica balanceada após a inserção sequencial da informação acima. Quais as operações de rotação que deveriam ser aplicadas após a inserção de cada um dos elementos acima, para que no final a estrutura de dados fique balanceada? Esquematize a estrutura binária balanceada final.

1c) (3 valores) Codifique uma função que conte o total de pessoas com um determinado primeiro nome. Por exemplo, para os dados acima e o nome "André", deveria devolver 2.

Protótipo: int totalNomeProprio(NodoAb *A, char *nome);

1d) (3 valores) Implemente uma função que imprima o nome de todos os elementos com código superior a determinado valor. Suponha que tem à sua disposição a função "**int contaNos(NodoAb *A)**", que retorna o total de nós de uma árvore binária.

Protótipo: void listaNomes(NodoAb *A, int cod);

1e) (3 valores) Implemente uma função que conte o total de nós a um determinado nível da árvore. Se achar conveniente, pode criar funções auxiliares com parâmetros adicionais.

Protótipo: int contaNosNivel(NodoAb *A, int nivel);

2. Considere a seguinte estrutura de dados, utilizada para representar um grafo com informação sobre as cidades de Portugal e as respetivas distâncias entre estas.

```
typedef struct ESTRADA{
        int idDestino;
        int distancia;     //valor em km.
        struct ESTRADA *nseg;
}Estrada;
```

2a) (3 valores) Implemente uma função que imprima os pares de IDs de cidades vizinhas. Considera-se que duas cidades são vizinhas quando estão a menos de "x" kms. uma da outra. Suponha que tem à sua disposição a função "**int** Dijkstra(Estrada **G, int tv, int v)**" que calcula o melhor caminho entre uma cidade "v" e os restantes elementos do grafo.

Protótipo: void listaVizinhas(Estrada **G, int tv, int x); //x=limite de kms.

2b) (4 valores) Implemente uma função que remova uma cidade do grafo, mantendo a consistência deste. Suponha que tem à sua disposição a função "**Estrada* apagaElemento(Estrada *L, Estrada *del)",** que remove o elemento "del" da lista "L".

Protótipo: void apagaCidade(Estrada ***G, int tv, int v); //v=cidade a remover