UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Educação e Ciência

U.C. 21046

Estruturas de Dados e Algoritmos Fundamentais

27 de junho de 2017

INSTRUÇÕES

- Leia estas instruções na totalidade antes de iniciar a resolução da prova.
- O enunciado da prova é constituído por 5 grupos de questões e termina com a palavra FIM.
- Se o seu exemplar não estiver completo ou nele se verificar qualquer outra deficiência, por favor dirija-se ao professor vigilante.
- A prova deve ser resolvida na sua totalidade em folhas de respostas.
- Nas respostas, tenha a preocupação de utilizar uma letra legível.
- Todas as respostas devem ser escritas unicamente com caneta azul ou preta.
- O teste é SEM CONSULTA. Todos os elementos necessários à resolução são fornecidos no enunciado.
- É permitido utilizar máquina de calcular.
- As cotações são indicadas por grupo e nas próprias questões.
- Nas questões de escrita de programas, a sua correção terá em conta critérios de proficiência e compreensibilidade do código (legibilidade, indentação, estrutura, comentários e explicação geral).
- O não cumprimento das instruções implica a anulação das respetivas questões.
- O tempo de realização da prova é de 150 minutos.

Grupo I [3 valores]

1.1. [1] Utilizando a definição, prove que $f(n) = n + \sqrt{n}$ é $\Omega(n)$.

1.2. Para cada um dos seguintes pares de funções $f(n)$ e $g(n)$, indique se $f(n) = O(g(n))$, $f(n) = \Omega(g(n))$, $f(n) = \Theta(g(n))$ ou nenhum dos casos.

1.2.1. [0.5] $f(n) = \sqrt{n}$, $g(n) = \log_2(n+2)$

1.2.2. [0.5] $f(n) = n^4 + n^2$, $g(n) = 2^n - n^2$

1.3. [1] Considere a complexidade do seguinte segmento de código em termos do nº $f(n)$ de operações aritméticas realizadas na variável a. Determine a expressão de $f(n)$ e indique a sua complexidade na notação $O(.)$.

```
for(a=0,i=1; i<=n; i*=2)
   for(j=1; j<=n; ++j)
      a++;
```

Grupo II [5 valores]

2.1. [1] Considere uma árvore de pesquisa binária inicialmente vazia na qual foram inseridas as chaves 9, 5, 4, 3, 7, 8, 11, 10, 12, 6 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções. Nas alíneas seguintes considere a árvore obtida como a árvore "original".

2.2. [1] Remova da árvore original a chave 5 utilizando o algoritmo de remoção por fusão (Deletion by Merging). Desenhe a árvore obtida.

2.3. [1] Efetue na árvore original uma rotação à esquerda de 11 em torno de 9. Desenhe a árvore obtida.

2.4. [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente vazia. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R): I 5, 20, 14, 22, 17, R 22 (total de 6 desenhos).

Grupo III [4 valores]

3.1. [2] Considere uma tabela T de dispersão (hash) com dimensão 11 e função de hash $h(x) = x \bmod 11$. As colisões são resolvidas com sondagem (probing) linear. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 1, 8, 21, 7, 18, 19, 20 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

3.2. [2] Considere o vetor [9 2 5 3 8 6 1 7 4]. Indique a sequência de passos para a sua ordenação utilizando o algoritmo Quicksort. Use como pivot o elemento do meio.

Grupo IV [4 valores]

4. Considere as seguintes classes para implementação de uma estrutura de dados tipo lista simplesmente ligada (Single Linked List) em que os itens são inteiros.

```
// definir nó
class ISLLNode {
public:
   // atributos
   int info;              // item
   ISLLNode *next;        // sucessor
};

// definir lista simplesmente ligada
class ISLL {
private:
   // atributos
   ISLLNode *head;       // primeiro nó
public:
   // construtores
   ISLL() {head=0;}      // cria lista vazia
   // métodos
   void insertHead(int x);
   void deleteTail();
    int delete(int x);
};
```

Na resolução das alíneas seguintes pode criar outros métodos e/ou construtores que achar convenientes. Explique em termos gerais o funcionamento do código e indique situações especiais ou casos particulares que necessitem de ser considerados.

4.1. [1] Implemente o método "void insertHead(int x)"que insere um item no início da lista.

4.2. [1] Implemente o método "void deleteTail()"que remove o último item da lista.

4.3. [2] Implemente o método "int delete(int x)"que remove a primeira ocorrência do item na lista. O método deve retornar 0 em caso de sucesso e -1 se o item não for encontrado.

Grupo V [4 valores]

5. Considere as seguintes classes para implementação de uma estrutura de dados tipo árvore de pesquisa binária (Binary Search Tree) em que os itens são genéricos.

```
// definir nó
template<class T>
class BSTNode {
public:
   // atributos
   T info;                  // item
   BSTNode *left, *right;   // filhos
};

// definir árvore BST
template<class T>
class BST {
private:
   // atributos
   BSTNode<T> *root;    // raiz
public:
   // construtores
   BST() {root= 0;}     // cria BST vazia
   // métodos
   void preorder();
    int deleteByCopying(const T& x);
};
```

Na resolução das alíneas seguintes pode criar outros métodos e/ou construtores que achar convenientes. Explique em termos gerais o funcionamento do código e indique situações especiais ou casos particulares que necessitem de ser considerados.

5.1. [1] Implemente o método "void preorder()"que efetua a travessia da árvore em pré-ordem (preorder). Visitar um nó significa imprimir o nó (atributo info).

5.2. [3] Implemente o método "int deleteByCopying(const T& x)"que remove um item da árvore utilizando o algoritmo de remoção por cópia. O método deve retornar 0 em caso de sucesso e -1 se o item indicado não for encontrado.

FIM