UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Educação e Ciência

U.C. 21046

Estruturas de Dados e Algoritmos Fundamentais

07 de julho de 2016

INSTRUÇÕES

- Leia estas instruções na totalidade antes de iniciar a resolução do teste.
- O enunciado do teste é constituído por 5 grupos de questões, tem 4 páginas 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.
- O teste deve ser resolvido 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 do teste é de 150 minutos.

Grupo I [3 valores]

1.1. [1] Utilizando a definição, prove que $f(n) = (n+1)^2$ é $O(n^2)$.

1.2. [1] 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. $f(n) = n + n\sqrt{n}$, $g(n) = 3n \log_2(n^2 + 1)$
2. $f(n) = n^4 - n^2$, $g(n) = 2^n + n$

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<=i; ++j)
      a++;
```

Grupo II [4 valores]

2.1. [2] Considere uma árvore de promoção (Splay Tree) inicialmente vazia na qual foram inseridos os itens 3, 1, 5, 4, 8, 12, 7 e seguidamente foram acedidos os itens 5, 4 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções e a seguir a cada acesso (total de 3 desenhos).

2.2. [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 3, 18, 12, 20, 15, R 3 (total de 6 desenhos).

Grupo III [5 valores]

3.1. [3] Considere uma tabela T de dispersão (hash) com dimensão 11 e função de hash $h(x) = x\%11$. As colisões são resolvidas com sondagem (probing) quadrática. Considerando a tabela inicialmente vazia, determine o conteúdo final da tabela após a inserção das chaves 3, 21, 17, 30, 12, 28, 13 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

3.2. [2] Considere o vetor [3 6 1 7 2 5 8 9 4]. Indique a sequência de passos para a sua ordenação utilizando o algoritmo Mergesort.

Grupo IV [4 valores]

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

```
// definir nó (duplamente ligado)
class IDLLNode {
public:
   // atributos
   int info;                // item
```

```
   IDLLNode *prev, *next;   // antecessor e sucessor
   // construtores
   IDLLNode() {prev=next=0;}
};

// definir lista duplamente ligada
class IDLL {
private:
   // atributos
   IDLLNode *head, *tail;   // primeiro e último nó
public:
   // construtores
   IDLL() {head= tail= 0;}  // cria lista vazia
   // métodos
   void insertHead(int x);
   void deleteTail();
    int deleteNode(int x);
};
```

Na resolução das alíneas seguintes pode criar outros métodos e/ou construtores que achar convenientes.

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 deleteNode(int x)"que remove a 1ª ocorrência do item na lista e retorna a sua posição ou -1 se não for encontrado. Considera-se que as posições começam em 0.

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;   // dois apontadores (filhos)
   // construtores
   BSTNode() {left=right=0;}
};

// definir árvore BST
```

```
template<class T>
class BST {
private:
   // atributos
   BSTNode<T> *root;        // um apontador (raiz)
public:
   // construtores
   BST() {root= 0;}   // cria BT vazia
   // métodos
   void insert(const T& x);
    int height();
};
```

Na resolução das alíneas seguintes pode criar outros métodos e/ou construtores que achar convenientes.

5.1. [1.5] Implemente o método "void insert(const T& x)"que insere um item na árvore. Explique em termos gerais o funcionamento do código.

5.2. [0.5] Apresente a definição de altura (height) de uma árvore. Indique a altura para os casos particulares de uma árvore vazia e uma árvore só com um nó.

5.3. [2] Implemente o método "int height()"que retorna a altura da árvore. Explique em termos gerais o funcionamento do código.

FIM