UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Ciência, Tecnologia e Ensino Superior

**U.C. 21046**

**Estruturas de Dados e Algoritmos Fundamentais**

**21 de junho de 2018**

## 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 $f(n) = n^2 + \log_2 n + 1$ é $O(n^2)$.

**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] $f(n) = n^2 + 1$, $g(n) = n\sqrt{n} + 10$

**1.2.2.** [0.5] $f(n) = 2^n + n^3 + 1000$, $g(n) = 3 \cdot 2^n + n^2 \log_2 n + 5$

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

## Grupo II [5 valores]

**2.1.** Considere uma árvore de promoção (Splay Tree) inicialmente vazia.

**2.1.1.** [1] Insira na árvore as chaves 8, 5, 10, 9, 12, 1, 7, 6 pela ordem indicada. Desenhe a árvore obtida após efetuadas todas as inserções. Nas alíneas seguintes considere a árvore obtida como a árvore "original".

**2.1.2.** [1] Remova da árvore original a chave 8 utilizando o algoritmo de remoção por cópia (Deletion by Copying) com a chave antecessora. Desenhe a árvore obtida.

**2.1.3.** [1] Considere que na árvore original foram efetuados acessos às chaves 7 e 10 pela ordem indicada. Desenhe a árvore obtida após cada acesso (total de 2 desenhos).

**2.2.** [2] Considere uma árvore B (B-Tree) de ordem 3 inicialmente contendo apenas o nó raiz com as chaves 8, 14. Desenhe a árvore após cada uma das seguintes operações de inserção (I) e remoção (R) pela ordem indicada: I 11, 18, 15, 9, 10; R 14; (total de 6 desenhos).

## Grupo III [4 valores]

**3.1.** [2] Considere uma tabela T de dispersão (hash) com dimensão 9 e função de hash $h(x) = x \bmod 9$. 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, 14, 12, 16, 11, 21, 20 pela ordem apresentada. Justifique os cálculos efetuados para cada inserção.

**3.2.** [2] Considere o vetor [5 8 9 4 3 6 1 7 2]. Indique a sequência de passos para a sua ordenação utilizando o algoritmo Mergesort. Justifique de um modo geral o funcionamento do algoritmo.

## Grupo IV [4 valores]

**4.** Pretende-se conceber em C++ uma estrutura de dados tipo pilha (Stack) implementada por uma lista simplesmente ligada (Single Linked List) em que os itens são inteiros. A pilha deve suportar as operações de inserir (push) e remover (pop) itens da pilha.

**4.1.** [1] Defina as classes que entender necessárias para a implementação da pilha. Defina apenas atributos e métodos, não inclua código para os métodos.

**4.2.** [1.5] Implemente o método "push"que insere um item no topo da pilha.

**4.3.** [1.5] Implemente o método "pop"que remove e retorna um item no topo da pilha. Admita que a pilha não está vazia.

## 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.

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

template<class T>   // definir árvore BST
class BST {
private:
   BSTNode<T> *root;    // raiz
public:
   void insert(const T& x);
    int height();
};
```

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.** [1.5] Implemente o método "void insert(const T& x)"que insere um item na árvore.

**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.

**FIM**