Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

Algoritmos e Estruturas de Dados

Exame Final – 2ª Parte – 12/JAN/2024

Atenção:

- As funções desenvolvidas devem ser apresentadas usando a **Linguagem C.**
- Não deve usar **variáveis globais.**

1 – Uma possível estratégia para efetuar a ordenação de um vector por Seleção Linear consiste basicamente, para cada passo, em procurar a primeira ocorrência do menor elemento de um (sub-)vector e trocá-lo depois com o primeiro. Considere, sem perda de generalidade, vectores cujos elementos são números inteiros.

a) Construa uma função **iterativa** que ordene um vector usando a estratégia indicada. A função deverá ter o seguinte protótipo:

```
void SelectionSort( int* a, unsigned int n );
```

b) Analise o número de **comparações** — associadas a elementos do vector — efectuadas pelo algoritmo desenvolvido.

Escreva uma expressão inicial usando somatórios e obtenha a expressão final para o número de comparações realizadas. Indique a ordem de complexidade do algoritmo.

c) Analise o número de trocas — associadas a elementos do vector — efectuadas pelo algoritmo desenvolvido.

Analise o comportamento do algoritmo no Melhor Caso, no Pior Caso e no Caso Médio. Indique configurações do vector que conduzam a esses casos.

Nota:

$$\sum_{k=1}^{n} \frac{1}{k} \approx \ln(n) + \gamma \quad \text{, com } \gamma = 0{,}577...$$

Cotação: a) – 2.0 b) – 1.0 c) – 1.0

2 – Considere uma estrutura de dados que permite a representação de uma sequência de números naturais, usando uma **lista ligada**, em que os sucessivos elementos se encontram **ordenados de modo não-decrescente**, existindo eventuais elementos **repetidos**.

O acesso à lista é feito a partir de um **nó cabeçalho**, que contém um ponteiro para o primeiro nó da lista e um valor inteiro que indica o número de elementos da sequência.

a) Dadas duas sequências — possivelmente vazias — representadas usando essa estrutura de dados, desenvolva uma função **iterativa** que devolve a **nova sequência** que resulta da sua **fusão ordenada**.

A função deverá ter o seguinte protótipo:

```
header_pointer Merge( header_pointer seq_1, header_pointer seq_2 );
```

b) Considere que as sequências dadas têm tamanhos n1 e n2 onde $n1 \geq 0$ e $n2 \geq 0$. Faça uma **análise da complexidade** do algoritmo anterior para o **Melhor Caso** e o **Pior Caso**, atendendo ao número de **comparações entre elementos das sequências**.

Cotação: a) – 2.0 b) – 1.0

3 – Considere o tipo abstrato de dados **Árvore Binária de Inteiros**, em cujos nós é possível armazenar um número inteiro. Considere que os elementos da árvore se encontram ordenados **(in-order)** e que não há elementos repetidos.

a) Desenvolva uma função **recursiva** que, dado o ponteiro para o nó raiz de uma árvore, conte o número de nós que **não são folhas** da árvore. A função deverá ter o seguinte protótipo:

```
unsigned int CountNonLeafNodes( pointer root );
```

b) Que tipo de travessia é efetuada pelo algoritmo da função anterior?

c) Desenvolva uma função recursiva eficiente — que **não aceda a nós desnecessários** — e que, dado o ponteiro para o nó raiz de uma árvore, conte o número de elementos pertencentes ao intervalo [a,b], b≥a.

A função deverá ter o seguinte protótipo:

```
unsigned int CountValuesInRange( pointer root, int a, int b );
```

Cotação: a) – 1.0 b) – 0.5 c) – 1.5