Departamento de Electrónica e Telecomunicações da Universidade de Aveiro

Algoritmos

Exame Final — 18 de Junho de 2003

1 – O Algoritmo ***"Bubble Sort"*** é um dos métodos possíveis para efectuar a ordenação de um vector e consiste, basicamente, em comparar, repetida e sucessivamente, elementos adjacentes de um (sub-)vector e realizar a sua *"troca"*, se necessário.

a) Construa uma função que percorra *uma vez* os elementos de um vector $v[0..n-1]$, $n > 1$, comparando e ordenando elementos adjacentes. A função deverá devolver um inteiro, 1 ou 0, indicando se foi ou não necessário *"trocar"* algum par de elementos adjacentes.

b) Faça uma *análise completa* do número de **comparações** e de **trocas** — entre elementos do vector — efectuadas pelo algoritmo da alínea anterior.

c) Construa agora outra função que, usando a função anterior, implemente o Algoritmo *"Bubble Sort"*.

d) Faça uma *análise completa* do número de **comparações** e de **trocas** — entre elementos do vector — efectuadas pelo algoritmo de ordenação da alínea anterior.

Nota:

$$\sum_{k=0}^{n} k^2 \;=\; \frac{1}{6}\, n\,(n+1)\,(2n+1)$$

2 – Considere o tipo abstracto de dados **Árvore Binária de Inteiros**, em cujos nós é possível armazenar um número inteiro.

Considere também que os números inteiros se encontram registados **"em-ordem" crescente**.

Elabore funções eficientes que permitam:

a) Determinar o número de **nós intermédios** — i.e., tendo pelo menos um descendente — de uma árvore dada.

b) Verificar se um dado número inteiro pertence a uma árvore e determinar o seu **nível** na árvore, caso lhe pertença.

c) Listar o **factor de equilíbrio** de cada um dos nós de uma dada árvore. O factor de equilíbrio de um nó é a diferença das alturas das suas duas sub-árvores.

v.s.f.f.

3 – Considere o tipo abstracto de dados **Grafo**, definido usando uma estrutura de dados dinâmica que representa um dado grafo $G(V, E)$, com n vértices e m arestas, armazenando a **lista dos vértices** do grafo e associando a cada elemento dessa lista (i.e., a cada vértice) a correspondente **lista de adjacências**.

Considere também que os n vértices de um grafo se encontram identificados pela sequência de números inteiros $0, 1, \ldots, (n-1)$.

Dado um vértice $v_i \in V$, pretende-se verificar se cada um dos outros vértices do grafo é ***"alcançável"*** a partir de v_i.

a) Desenvolva uma função que, usando uma estratégia **recursiva**, verifique se todos os outros vértices são "alcançáveis" a partir de um dado vértice de um grafo.

b) Desenvolva agora uma função que, usando uma estratégia **iterativa**, verifique se todos os outros vértices são "alcançáveis" a partir de um dado vértice de um grafo.

Sugestões:

— Use uma travessia apropriada.

— Não se esqueça de que o grafo pode conter ciclos.

— Assuma que estão definidos os tipos abstractos **Pilha** e **Fila**; não é necessário implementá-los.