Estruturas de Dados e Algoritmos II
Exame de Recurso

Departamento de Informática
Universidade de Évora

22 de Junho de 2017

1. [2,5 valores] Assumindo que o alfabeto consiste nas 26 letras minúsculas, desenhe uma *trie* cujo conteúdo sejam as quatro palavras

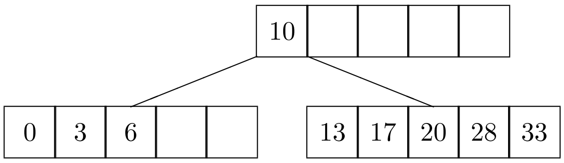
acto   facto   fartote   farto

Qual seria a memória ocupada por uma implementação em C da *trie* que desenhou, numa máquina com endereços e palavras de 32 bits? (Não precisa de calcular o valor, mas apresente e justifique todos os cálculos efectuados ou a efectuar.)

2. [2,5 valores] A *B-tree* da figura tem grau de ramificação mínimo 3. Apresente o seu estado depois da execução de *cada uma* das operações da sequência

i 18   **r** 10   **r** 17   **i** 36   **r** 6   **i** 6   **r** 28   **r** 33

pela ordem apresentada. As letras **i** e **r** indicam, respectivamente, a inserção e a remoção do elemento que se lhes segue.



3. [2,5 valores] Seja o valor de uma sequência a soma dos valores dos seus elementos. Dadas duas sequências de inteiros não negativos,

$$V = v_1, v_2, \ldots, v_n \;\text{ e }\; S = s_1, s_2, \ldots, s_n, \;\text{ para algum } n \geq 1,$$

pretende-se obter uma subsequência de V de valor máximo, sujeita à seguinte regra: se o elemento v_i pertencer à subsequência, os s_i elementos de V que se seguem ao elemento v_i não podem pertencer-lhe. (Por outras palavras, se o elemento v_i for escolhido para a subsequência, então é necessário saltar os próximos s_i elementos antes de escolher um novo elemento para a subsequência.)
Por exemplo, dadas as sequências abaixo, se o elemento 3, cujo valor é 5, pertencer à subsequência, os 2 elementos seguintes (índices 4 e 5), não poderão pertencer-lhe. Neste caso, a subsequência com maior valor contém os elementos 2, 4 e 5, cujos valores somam 9.

Índice	1	2	3	4	5	6
Valor (V)	1	2	5	4	3	1
Saltar (S)	2	0	2	0	2	2

Apresente *uma função recursiva* que, dadas duas sequências de inteiros, V e S, calcula o valor máximo possível para uma subsequência de V construída de acordo com a regra enunciada.
Indique claramente o que representa cada uma das variáveis que utilizar e explicite a chamada inicial. (Note que não é pedido que escreva código.)

4. [2,5 valores] Considere a função recursiva $F_X[i, j]$, onde:

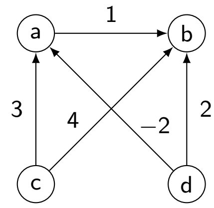
- n é um inteiro positivo; e
- $X = (x_{ij})$, para $1 \leq j \leq i \leq n$, é uma sequência não vazia de inteiros.

$$F_X[i,j] = \begin{cases} 0 & \text{se } i = n+1 \\ x_{ij} + \max\{\ F_X[i+1,j],\ F_X[i+1,j+1]\ \} & \text{se } i < n \end{cases}$$

Apresente o pseudo-código de um algoritmo iterativo que, dada uma sequência $X = (x_{ij})$ não vazia de inteiros, com $1 \leq j \leq i \leq n$, calcula e devolve o valor de $F_X[1, 1]$.

5. Seja DAG o algoritmo seguinte, onde $G = (V, E)$ é um grafo pesado orientado, w é a função de peso do grafo e S é uma ordenação topológica dos nós de G, e seja G_5 o grafo da figura.

```
DAG(G, w, S)
1.  n ← |G.V|
2.  let x be uma matriz com dimensões [1..n,1..n]
3.  for each nó u ∈ G.V do
4.      x[u, u] ← 0
5.      for each nó v ∈ G.V − {u} do
6.          x[u, v] ← ∞
7.  for each nó u ∈ G.V pela ordem S do
8.      for each arco (u,v) ∈ G.E do
9.          x[u,v] ← w(u,v)
10.         for each nó t < u na ordem S do
11.             if x[t,u] + x[u,v] < x[t,v] then
12.                 x[t,v] ← x[t,u] + x[u,v]
13. return x
```



Grafo G_5

(a) [1 valor] Diga, justificando, se G_5 é fortemente conexo.

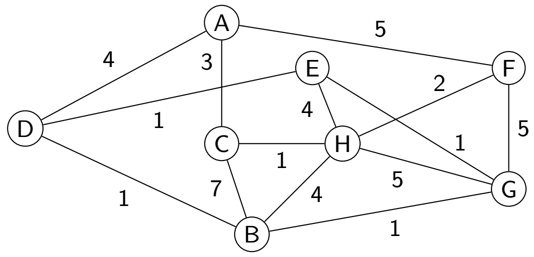
(b) [1 valor] Apresente uma componente fortemente conexa de G_5.

(c) [1 valor] Apresente uma ordenação topológica para os nós de G_5.

(d) [1,5 valores] Sendo o grafo representado através da listas de adjacências, analise a complexidade temporal do algoritmo DAG no pior caso.

(Assuma que a selecção de *cada nó*, feita nas linhas 7 e 10, tem custo constante.)

6. Seja G_6 o grafo representado na figura abaixo.



(a) [1,5 valores] Apresente uma árvore de cobertura mínima para G_6. Qual o peso dessa árvore?

(b) [1,5 valores] Apresente uma ordem pela qual os vértices poderiam ser incluídos na árvore durante a sua construção pelo algoritmo de Prim, a partir do vértice F.

7. [2,5 valores] As tarefas complexas realizam-se através da realização de subtarefas mais simples. Por exemplo, para a construção de um edifício é necessário desenhá-lo, adquirir o terreno, obter as licenças de construção, fazer as fundações, etc., etc., até à pintura das paredes, à colocação dos azulejos, à instalação das portas, ...

A realização de muitas das subtarefas depende da realização prévia de outras subtarefas, mas algumas podem ser executadas em paralelo. Por exemplo, para obter as licenças de construção é necessário ter o projecto do edifício, mas não é necessário que este esteja elaborado para proceder à compra do terreno. O conhecimento das dependências entre subtarefas e do tempo estimado para a realização de cada subtarefa, permite estimar a duração da tarefa principal, mas o cálculo dessa estimativa pode ser demasiado complexo para fazer à mão se a quantidade de subtarefas e de dependências entre elas for muito grande.

Se lhe fossem dadas todas as subtarefas a executar e, para cada uma, a sua duração estimada e a lista de subtarefas que dela dependem, como poderia calcular a estimativa para a duração mínima da tarefa principal? Descreva detalhadamente como modelaria os dados e como os representaria, diga que algoritmo(s) utilizaria, porquê, e como obteria o resultado pretendido.

(Assuma que a duração estimada de qualquer subtarefa é positiva e que não há dependências cíclicas entre subtarefas.)