DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

AED - 2012-2013 - 1º Semestre
Algoritmos e Estruturas de Dados

1^o Exame, 12 Janeiro 2012, 11:30h   Duração: 3 horas

Prova escrita, individual e sem consulta

NOME:______________________________ NÚMERO:____________

PARTE I - Questões de Escolha Múltipla

Preencha as respostas na tabela (usando apenas letras maiúsculas). Se nenhuma opção servir, escreva **NENHUMA**. Se pretender alterar a sua resposta, risque e escreva ao lado a sua nova opção. Todas as questões de escolha múltipla valem 0.75 valores. As questões de escolha múltipla não respondidas são cotadas com 0 valores, mas por cada resposta errada são descontados 0.75/4 valores.

Questão	1	2	3	4	5	6	7	8
Resposta								

1. Indique qual das seguintes afirmações é **verdadeira**:

A. $g(N) \in \mathcal{O}(f(N))$ *sse* $\forall c_0 \in \mathbb{R}^+, N_0 \in \mathbb{N}, \exists N \geq N_0 : 0 \leq g(N) \leq c_0\, f(N)$

B. $g(N) \in \mathcal{O}(f(N))$ *sse* $\exists c_0 \in \mathbb{R}^+, N_0 \in \mathbb{N} : 0 \leq g(N) \leq c_0\, f(N), \forall N \geq N_0$

C. $g(N) \in \mathcal{O}(f(N))$ *sse* $\exists c_0 \in \mathbb{R}^+ : 0 \leq g(N) \leq c_0\, f(N), \forall N \in \mathbb{N}$

D. $g(N) \in \mathcal{O}(f(N))$ *sse* $\forall c_0 \in \mathbb{R}^+, \exists N_0 \in \mathbb{N} : 0 \leq g(N) \leq c_0\, f(N), \forall N \geq N_0$

2. Considere o seguinte trecho de código à esquerda. Indique qual dos conjuntos à direita melhor reflete a complexidade computacional do mesmo:

```
...
for (i = 0; i < N; i++) {
   j = N-1;
   while (j > 0) {
      A[i][j] = U[i] * V[j];
      j = j / 2;
      if ((i == 0) && (j == 0))
         A[i][j] = exp(w * U[i]);
   }
}
...
```

A. $\mathcal{O}(N^2)$

B. $\mathcal{O}(N \log^2 N)$

C. $\mathcal{O}(N \log N)$

D. $\mathcal{O}(N)$

3. Após a aplicação de um determinado número de iterações do **algoritmo da união rápida ponderada (equilibrada)**, no problema da conectividade, obteve-se a seguinte tabela

0	1	2	3	3	5	3	5	5	3

Qual a tabela que se obtém se, de seguida, se ligar o par 4-5?

A.

0	1	2	3	3	3	3	5	5	3

B.

0	1	2	5	3	5	3	5	5	3

C.

0	1	2	3	3	4	3	5	5	3

D.

0	1	2	3	5	5	3	5	5	3

4. Considere os algoritmos básicos de ordenação estudados (seleção, inserção e *bubble*). Indique qual das seguintes afirmações é **verdadeira**. Assuma em cada caso a melhor implementação de cada algoritmo (adaptativa, optimizada, etc):

   A. Quando os dados estão já ordenados ou quase ordenados, o algoritmo com pior desempenho assintótico é o *Bubble*.

   B. Quando os dados estão já ordenados ou quase ordenados, o algoritmo com melhor desempenho assintótico é o *Seleção*.

   C. Quando os dados estão já ordenados ou quase ordenados o algoritmo com pior desempenho assintótico é o *Inserção*.

   D. Quando os dados estão já ordenados ou quase ordenados o algoritmo com pior desempenho assintótico é o *Seleção*.

5. Uma árvore diz-se balanceada AVL se e só se, em todos os nós, a diferença entre as alturas das suas sub-árvores for igual ou inferior a um. Com base nesta propriedade diga qual das seguintes estruturas **não pode** representar uma árvore AVL?

   A. Uma árvore vazia. B. Uma árvore com três nós e altura um.
   C. Uma árvore com dois nós. D. Uma árvore com três nós e altura dois.

6. Considere dois vetores, `v1` e `v2`, em que cada um armazena `N` inteiros e que estão ambos ordenados por ordem **crescente**, sendo que `v1[N-1]` < `v2[0]`. Suponha que é feito o seguinte procedimento: (i) começando na posição `0`, é retirado sucessivamente um elemento de `v1` seguido de outro de `v2` que são colocados num **acervo** (em que a prioridade é inversa do valor lá colocado, ou seja quanto menor o número maior a prioridade); (ii) depois de esvaziar os vetores para o acervo, são feitas `N` chamadas a `RemoveMax()` retirando em cada uma o elemento mais prioritário do acervo, repondo a condição de acervo e colocando o número retirado no vector `v1`, sequencialmente, a partir da posição `0`; (iii) são feitas mais `N` chamadas a `RemoveMax()` e os valores retirados do acervo são agora colocados em `v2` sequencialmente, a partir da posição `0`. Diga qual das afirmações seguintes é verdadeira:

   A. No final `v1` fica com os valores inicialmente colocados em `v2` na mesma ordem e vice-versa.

   B. Os vetores `v1` e `v2` ficam com os mesmos valores, na mesma posição que tinham inicialmente.

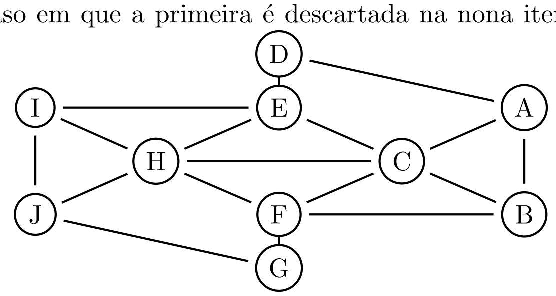
   C. Os vetores `v1` e `v2` ficam com os mesmos valores que tinham inicialmente mas ordenados por ordem inversa.

   D. No final os valores inicialmente contidos em `v1` e `v2` ficam arbitrariamente distribuídos pelos dois vetores.

7. Suponha que os seguintes números são introduzidos numa tabela de dispersão ("hash table"): **314**, **419**, **14**, **403**, **214**, **203**, **14**, **111**. Assuma que a função de dispersão (para indexação na tabela) é a seguinte: $f(x) = x \; mod \, 20$ (ou seja, o resto da divisão por 20). Assuma que números repetidos não são armazenados, quando tal é verificado, e que colisões são resolvidas por lista, com inserção no **final**. Qual das afirmações é **verdadeira**? (contabilize comparações apenas entre números)

   A. Ocorrem 3 colisões e 5 comparações. B. Ocorrem 3 colisões e 6 comparações.
   C. Ocorrem 4 colisões e 6 comparações. D. Ocorrem 4 colisões e 5 comparações.

8. Considere o grafo indicado em baixo à esquerda e assuma que o mesmo não é direcionado mas é ponderado, como se indica do lado direito do grafo. Se aplicar o algoritmo de Kruskal para obtenção da Árvore Mínima de Suporte ("MST") e os pares de arestas indicados abaixo, qual corresponde ao caso em que a primeira é descartada na nona iteração e a segunda entra na décima iteração?



A↔D,	B↔F,	E↔H	2
A↔B,	C↔F		3
A↔C,	D↔E,	G↔J	4
B↔C,			5
E↔I			6
C↔H,	I↔J	C↔E	7
H↔I,	F↔H		8
H↔J,	F↔G		9

A. (B – C); (E – I). B. (B – C); (C – E). C. (G – J); (B – C). D. (B – C); (I – J).

PARTE II - Questões de Desenvolvimento

Responda a cada uma das questões de desenvolvimento em **folhas de exame separadas** e devidamente identificadas com nome e número.

[4.5] 9. Pretende-se desenvolver uma solução eficiente para um problema computacional envolvendo um vetor `a[...]` de dimensão N. De seguida descreve-se, à esquerda, o problema e, à direita, a solução proposta (em que se assume que a matrix bidimensional `M` foi inicializada a `0`).

Dado um vetor `a[]` de dimensão N, pretende-se obter uma matriz M de dimensão $N \times N$ tal que:

$$\begin{cases} M(i,j) = \sum_{k=0}^{j} a[k], & i \leq j \\ M(i,j) = 0, & i > j \end{cases}$$

```
void calc_M(int *a, int **M, int N)
{
  int i, j, k, soma;

  for(i = 0; i < N; i++)
    for(j = i; j < N; j++) {
      soma = 0;
      for(k = 0; k <= j; k++)
        soma += a[k];
      M[i][j] = soma;
    }
  return;
}
```

[1.0] a) Indique, em função de N, qual a complexidade computacional da solução proposta.

[3.0] b) Desenvolva em C uma solução mais eficiente para o mesmo problema.
Nota: a sua solução tem de ter uma complexidade assintótica melhor que a apresentada!
Sugestão: Construa a matriz `M` para o vetor `a`= [1 2 3 4] e repare na estrutura da matriz resultante.

[0.5] c) Determine em função de N a complexidade computacional da solução apresentada em 9-b).

[3.5] 10. Considere de novo o grafo do Problema 8 no topo desta página:

[3.0] a) Tomando o vértice A como ponto de partida, determine a Árvore de Caminhos mais Curtos ("SPT"). Admita que os nós são, em cada passo, examinados pela ordem alfabética. Indique justificadamente todos os passos que executar e descreva em detalhe os cálculos realizados. Indique também que algoritmo usou.

[0.5] b) Assumindo que possui um grafo representado por matriz de adjacências, discuta justificadamente qual a complexidade temporal do algoritmo de Dijkstra.

[3.0] 11. Considere um grafo com V vértices e E arestas cuja informação (de conectividade, eventualmente pesos de arestas, etc), está guardada numa dada representação.

[1.0] a) Discuta, em função das características do grafo, nomeadamente a relação entre V e E, qual a melhor representação a utilizar.

Indique justificadamente qual o algoritmo a usar se pretender resolver cada um dos seguintes problemas:

[1.0] b) Dados dois nós quaisquer do grafo, encontrar um caminho no grafo entre esses dois nós, qualquer que seja esse caminho.

[1.0] c) Dados dois nós quaisquer do grafo, encontrar o caminho mais curto entre eles.

[3.0] 12. Suponha que pretende armazenar `N` números inteiros que serão acedidos múltiplas vezes (assuma que `N` é muito grande). Para resolver o problema é-lhe proposta uma solução suportada numa implementação de uma tabela de dispersão, de dimensão `M`, que se baseia no <u>método de dispersão por índices livres</u> sendo as colisões resolvidas por procura linear. A proposta inclui igualmente a utilização da seguinte função de dispersão:

$$f(x) = 2 \times (x\%M)$$

[1.0] a) Analise se a opção escolhida para a função de dispersão é boa ou má e porquê. Consegue pensar numa solução melhor? Se sim, qual?

[1.0] b) Para que a solução possa funcionar, que relação tem necessariamente de existir entre `M` e `N`? Como pode garantir que nesse caso é sempre possível armazenar e encontrar um elemento na tabela?

[1.0] c) Discuta eventuais problemas que a solução proposta (i.e. o tipo de tabela de dispersão) possa ter e de que forma limitam o desempenho das operações de procura que pretende fazer?