DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

AED - 2016-2017 - 2º Semestre
Algoritmos e Estruturas de Dados

2º Exame, 6 de Julho de 2017, 18h30m Duração: 3 horas

Prova escrita, individual e sem consulta

NOME:_____ NÚMERO:_____

PARTE I - Questões de Escolha Múltipla (A/B/C/D)

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 seguintes valem 0.75 valores. Estas 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
Resposta						

1. Considere as implementações da função factorial $N!$, para $N > 0$ com $0! = 1$:

```
int factorial(int N)
{
  if (N == 0) return 1;
  return N*factorial(N-1);
}
```

```
int factorial(int N){
  int i, t;
  for (t = 1, i = 1; i <= N; i++) t *= i;
  return t;}
```

Qual das seguintes afirmações é verdadeira:

A. A implementação à esquerda possui complexidade $\mathcal{O}(\lg N)$;
B. O algoritmo da direita não representa a função factorial;
C. A implementação da esquerda é mais eficiente, já que usa recursividade;
D. Ambos requerem a mesma ordem de operações básicas.

2. Interpretando p como a operação abstracta `Push` e P como a operação abstracta `Pop`, ambas definidas para o tipo abstracto Pilha, qual das tabelas abaixo, iniciando processamento no início da tabela, produz a palavra `exames` quando submetida à sequência `pppPppPPpPppPPppp` para uma pilha inicialmente vazia.

A. | x | s | e | a | x | m | s | e | x | a | m |

B. | m | s | e | a | x | m | e | s | x | a | m |

C. | a | s | e | m | x | a | s | e | x | a | m |

D. | m | s | e | a | x | n | s | e | x | a | m |

3. Para um dado algoritmo recursivo determinou-se que a sua complexidade temporal pode ser descrita pela recorrência $C_N = C_{N-1} + \log(N^{2/3})$. Qual das funções abaixo é o menor dos majorantes que define a complexidade desse algoritmo?

A. $\log(N^{2/3})$ B. $N + \log(N)$ C. $N \log(N)$ D. $N \log^2(N)$

4. Considere a seguinte tabela de conectividade, obtida através do algoritmo de procura rápida:

$id[.] =$ | 0 | 1 | 1 | 7 | 4 | 4 | 6 | 7 | 8 | 9 |

ao qual se pretendem adicionar as ligações em baixo, pela ordem indicada:

$$(2,5), \quad (1,8), \quad (6,4), \quad \text{e} \quad (8,9).$$

Indique qual o número de alterações necessárias, a realizar na tabela de conectividade, quando se pretendem incluir estas quatro novas conexões.

A. 6	B. 7	C. 11	D. 13

5. Considere a implementação do *Quick Sort* à direita, para a ordenação de um vector `int a[]`, com limites à esquerda e direita identificados pelas variáveis `int l` e `int r`, respetivamente.

```
void quicksort(item a[], int l, int r)
{
  int i;
  if (r <= l) return;
  i = partition(a, l, r);
  quicksort(a, l, i-1);
  quicksort(a, i+1, r);
}
```

Para a seguinte tabela de inteiros:

10	4	15	6	3	5	9	7	8	15

Indique qual a sequência de resultados, para a variável `i`, ao longo das 4 quatro primeiras chamadas à função `quicksort`.

A. {8, 4, 3, 0}	B. {8, 4, 5, 7}	C. {1, 8, 5, 4}	D. {9, 4, 3, 0}

6. Considerando um grafo de 11 vértices, numerados de 0 a 10, uma das quatro tabelas abaixo não pode representar a árvore de procura final, através do vector (`st`), contendo a MST produzida pelo algoritmo de Prim. Qual?

A.

1	2	7	7	5	7	1	7	5	2	3

B.

9	3	6	9	4	3	4	3	6	4	9

C.

9	7	1	9	9	2	2	6	0	9	0

D.

1	8	2	2	2	2	5	3	2	1	5

PARTE II - Questões de Escolha Binária (V/F)

Preencha as respostas na tabela (usando apenas letras maiúsculas – **V**(erdadeira) ou **F**(alsa)). Todas as questões de escolha múltipla seguintes valem 0.50 valores. Estas questões de escolha múltipla não respondidas ou erradas são cotadas com 0 valores.

Questão	7	8	9	10	11	12	13	14
Resposta								

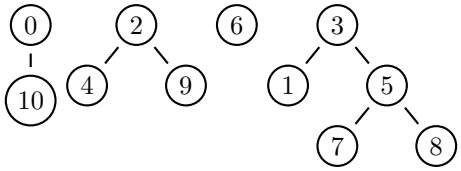
7. A procura em tabelas ordenadas é, em média, mais eficiente que a procura em árvores binárias ordenadas, quando as árvores são balanceadas AVL.

8. A existência de alguns vértices isolados não impede que o grafo a que pertençam seja denso.

9. Quando um problema só admite ser recursivamente decomposto em sub-problemas não independentes, a programação dinâmica ascendente é uma das estratégias de projecto adequadas.

10. Um algoritmo para grafos representados através de matriz de adjacências diz-se linear quando o seu tempo de execução é proporcional ao número de vértices.

11. As árvores à direita, não podem representar o resultado da conexão usando o método de procura rápida.



12. No pior dos casos, o *Quick Sort* necessita de tantas comparações quanto o *Selection Sort*.

13. A única implementação de filas possível, requere uma estrutura de dados com um ponteiro a apontar para o primeiro elemento, e outro a apontar para o último (como apresentado à direita).

```
struct fila
{
  element * primeiro;
  element * ultimo;
};
```

14. Um tipo abstracto é um tipo de dados genérico, dos quais não se conhecem os valores.

PARTE III - 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] 15. Para um grafo ponderado direccionado com 14 vértices, identificados de 0 a 13, aplicou-se o algoritmo de Dijsktra para determinação da árvore de caminhos mais curtos (SPT) usando o método tabular. A tabela completa está abaixo.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	∞	-	∞	6/1	18/1	24/1	∞	∞	∞	30/1	∞	∞	21/1	∞
3	∞	-	∞	-	18/1	7/3	17/3	∞	∞	26/3	∞	∞	21/1	∞
5	20/5	-	∞	-	18/1	-	17/3	9/5	∞	26/3	∞	23/5	21/1	∞
7	20/5	-	∞	-	18/1	-	17/3	-	∞	26/3	∞	13/7	19/7	23/7
11	20/5	-	33/11	-	18/1	-	17/3	-	∞	26/3	23/11	-	19/7	15/11
13	20/5	-	16/13	-	18/1	-	17/3	-	∞	26/3	23/11	-	19/7	-
2	20/5	-	-	-	18/1	-	17/3	-	36/2	26/3	23/11	-	19/7	-
6	20/5	-	-	-	18/1	-	-	-	32/6	24/6	23/11	-	19/7	-
4	20/5	-	-	-	-	-	-	-	26/4	24/6	23/11	-	19/7	-
12	20/5	-	-	-	-	-	-	-	26/4	24/6	23/11	-	-	-
0	-	-	-	-	-	-	-	-	25/0	24/6	23/11	-	-	-
10	-	-	-	-	-	-	-	-	25/0	24/6	-	-	-	-
9	-	-	-	-	-	-	-	-	25/0	-	-	-	-	-
8	-	-	-	-	-	-	-	-	-	-	-	-	-	-

[1.00] a) Trace a árvore que resultou da aplicação do algoritmo. Justifique as suas conclusões.

[1.00] b) O grafo dado possui uma aresta de fonte no vértice 11 e destino no vértice 9. Qual o custo mínimo que essa aresta possui. Justifique.

[1.25] c) Caso a SPT o permita, indique quais os vértices que fazem parte do caminho mais curto entre o vértice 3 e o vértice 8 e qual o seu custo. Se a SPT não permitir determinar esse caminho, justifique porquê.

[1.25] d) Proponha a adição de uma aresta que permita reduzir o custo do caminho mais curto entre o vértice 5 e o vértice 10 em duas unidades, bem como alterar alguns dos vértices que compõem o caminho, relativamente ao original, para chegar de 5 a 10. Para isso, primeiro identifique o caminho, usando a SPT, e determine o seu custo. De seguida indique, de forma justificada, qual a aresta a adicionar e qual o seu custo.

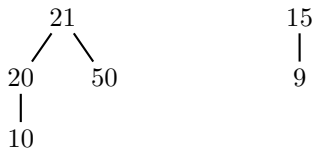
[2.0] 16. Considere a seguinte sequência de inteiros, que representam a inserção e remoção de elementos numa árvore binária vazia:

`I:20 I:15 I:50 I:10 I:9 R:50 I:21`,

em que `I:X` e `R:X` representam, respectivamente, a inserção e remoção do inteiro `X`. Durante a inserção e remoção de elementos, deve garantir que a árvore se mantém ordenada e balanceada.

[1.00] a) Indique o estado da árvore após cada operação de inserção/remoção justificando, se necessário, a aplicação de rotações na árvore de modo a manter as árvores balanceadas.

[1.00] b) Suponha que os elementos indicados, em cima, estão incluídos nas seguintes árvores:



Qual o resultados da junção das árvores indicadas, mantendo a árvore resultante ordenada e balanceada. Justifique todos os passos necessários.

[5.0] 17. Neste problema, pretende-se que os alunos implementem um algoritmo de cálculo do determinante de uma matriz quadrada $N \times N$. Para isso, é pedido que escrevam um código para resolver o *Teorema de Laplace*. Segundo este teorema, uma matrix $\mathbf{A}$ de dimensão $N \times N$ tem determinante:

$$\det(\mathbf{A}) = a_{i,1}\mathbf{C}_{i,1} + a_{i,2}\mathbf{C}_{i,2} + \cdots + a_{i,N}\mathbf{C}_{i,N}$$

em que $a_{i,j}$ corresponde ao elemento da linha i e coluna j (de notar que i é fixo e qualquer), e

$$\mathbf{C}_{i,j} = (-1)^{i+j}\det(\mathbf{A}_{i,j}).$$

em que $\mathbf{A}_{i,j}$ é uma submatrix de $\mathbf{A}$, obtida pela remoção das linhas i e coluna j.
Como exemplo de aplicação, considere uma matriz 4×4, e a escolha de $i = 1$:

$$\det\left(\begin{bmatrix}1&3&4&1\\7&3&4&1\\9&6&3&1\\2&2&7&3\end{bmatrix}\right) = 1 \cdot \underbrace{\det\left(\begin{bmatrix}3&4&1\\6&3&1\\2&7&3\end{bmatrix}\right)}_{(\mathcal{A})} - 3 \cdot \det\left(\begin{bmatrix}7&4&1\\9&3&1\\2&7&3\end{bmatrix}\right) + \ldots$$

$$(\mathcal{A}) \equiv \det\left(\begin{bmatrix}3&4&1\\6&3&1\\2&7&3\end{bmatrix}\right) = 3 \cdot \underbrace{\det\left(\begin{bmatrix}3&1\\7&3\end{bmatrix}\right)}_{=3\cdot3-1\cdot7} - 4 \cdot \underbrace{\det\left(\begin{bmatrix}6&1\\2&3\end{bmatrix}\right)}_{=6\cdot3-1\cdot2} + 1 \cdot \underbrace{\det\left(\begin{bmatrix}6&3\\2&7\end{bmatrix}\right)}_{=6\cdot7-3\cdot2}$$

...

Por forma a resolver este problema, deve utilizar a seguinte estrutura de dados:

```
struct _t_matrix {
  float **values;
  int N;
} t_matrix;
```

Nota: A cotação atribuída, em cada uma das alíneas abaixo, depende da eficiência dos algoritmos propostos, pelo que deve tentar produzir os algoritmos mais eficientes possíveis, não bastando apenas propor algoritmos que resolvam o problema.

[1.0] a) Escreva, em Linguagem C, as funções de alocação e libertação de memória:

`t_matrix * newMatrix( int N );` e `void freeMatrix( t_matrix * aM );`

para a estrutura de dados `t_matrix`.

[1.5] b) Escreva, em Linguagem C, a função:

`t_matrix * newSubMatrix( t_matrix *aM, int i, int j );`

que, a partir de uma matrix $\mathbf{A}$, $N \times N$, aloca e devolve a matrix $\mathbf{A}_{i,j}$, $N-1 \times N-1$, obtida removendo a linha i e coluna j de $\mathbf{A}$.

[2.5] c) Escreva, em linguagem C, uma função:

`float det( t_matrix* mA );`

que implemente o algoritmo por si proposto. A função deve devolver o determinante da matriz. Indique, justificando, a recorrência do algoritmo proposto.