DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

AED - 2017-2018 - 1º Semestre
Algoritmos e Estruturas de Dados

1^o Exame, 10 de janeiro de 2018, 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 o seguinte troço de código

```
int foo(int N) {
  int result = 0;
  int j,k;

  for (j = N; j > 0; j--) {
    for (k = 1; k < j; k+=k) {
        result += (j+k)/N;
    }
  }
  return result;
}
```

Qual dos seguintes conjuntos representa a complexidade da função `foo(.)`?

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

2. Aplicou-se a uma árvore binária um varrimento pós-fixado e um varrimento in-fixado, obtendo-se os seguintes resultados:
Pós-fixado: 3-4-5-2-7-8-6-9-1
In-fixado : 3-5-4-2-1-7-9-8-6

Qual dos resultados abaixo indicados poderia ser obtido com um varrimento em **largura** aplicado numa das suas sub-árvores.

Sugestão: Comece por traçar a árvore a partir da informação dada inicialmente.

A. 2-5-3-4	B. 2-5-4-3	C. 2-4-5-3	D. 4-2-5-3

3. Para ordenar a tabela abaixo, um dos algoritmos necessita de fazer apenas 8 trocas. Qual dos algoritmos é?

22	17	13	5	3	6	8

A. *QuickSort*	B. *InsertionSort*	C. *BubbleSort*	D. *SelectionSort*

4. Seja um algoritmo recursivo em que a memória alocada durante a sua execução pode ser expressa pela recorrência $M_N = 4M_{N/3} + 1$.Qual o conjunto que melhor descreve (menor majorante) a memória utilizada por esse algoritmo?

A. $\mathcal{O}(N^{\lg_3(4)} \lg(N))$	B. $\mathcal{O}(N^{\lg_4(3)})$	C. $\mathcal{O}(N^{\lg_3(4)})$	D. $\mathcal{O}(N^2)$

5. Suponha uma *hash table* com dimensão $M = 31$ na qual são adicionados elementos através de uma função de dispersão que é baseada na operação $x\%M$ (resto da divisão por M). Em determinada altura verifica-se que a tabela está cerca de 75% cheia e embora se queira usar pouca memória, decide-se construir uma tabela nova e fazer *rehash* dos dados para colocar na nova tabela de forma a esta estar menos de 50% cheia. De forma a garantir que o número de colisões se mantém semelhante, qual é um valor adequado para a nova dimensão M' da tabela?

A. $M' = 41$	B. $M' = 49$	C. $M' = 62$	D. $M' = 53$

6. Considere o seguinte vector de nós $id[\cdot]$ que representa os nove nós do problema. Qual a quantidade de atribuições $id[x] = y$ efectuadas ao aplicar o algoritmo Quick Find, para o ficheiro de entrada abaixo?

$id =$

1	2	3	4	5	6	7	8	9

Ficheiro de entrada:

$$(2,5), (8,7), (6,1), (2,7), (8,5), (3,7)$$

A. 6	B. 4	C. 8	D. 10

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. O algoritmo de ordenação **Selection** não é estável.
8. O varrimento in-fixado de um acervo produz um output ordenado, tal como em árvores ordenadas.
9. É possível construir uma estrutura de dados `S`, para elementos de um dado tipo, de forma a que as operações `Inserir(x,S)`, `Apagar(x,S)` e `Procurar(x,S)` têm todas um tempo médio de execução $\mathcal{O}(1)$.
10. Usando dispersão com o método de índices livres permite reduzir o número de colisões relativamente à utilização de dispersão por listas, assumindo a mesma função de dispersão.
11. Seja T uma árvore mínima de suporte de um grafo G. O caminho em T entre qualquer par de vértices u e v é sempre igual ao caminho mais curto entre u e v em G.
12. No algoritmo Quick Union se dois objectos apontarem para o mesmo elemento então estão ligados.
13. Se numa árvore binária ordenada forem introduzidos, por ordem crescente, os mil primeiros inteiros, a árvore resultante pode ter uma altura da ordem da dezena.
14. Uma implementação eficiente de BFS em grafos requer a utilização de uma estrutura adicional com disciplina **FIFO**, enquanto que a DFS requer uma estrutura com disciplina **LIFO**.

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.

[5.0] 15. Dada uma tabela com N números inteiros positivos pretende-se saber se existe algum valor "predominante" e qual é esse valor. Define-se como valor "predominante" qualquer valor que aparece tantas ou mais vezes na tabela que qualquer dos outros valores na tabela (se existir mais que um valor predominante, qualquer um deles pode ser indicado)

Um algoritmo possível para resolver este problema pode ser o seguinte:

```
int pred(int a[], int N)
{
  int i, j, val, val_max_freq, cnt, cnt_max;

  val_max_freq = a[0];
  cnt_max = 1;
  for (i=0; i<N-1; i++) {
    val = a[i];
    cnt = 1;
    for (j=i+1; j<N; j++) {
      if (val == a[j]) {
        cnt++;
      }
    }
    if (cnt > cnt_max) {
      cnt_max = cnt;
      val_max_freq = val;
    }
  }
  return val_max_freq;
}
```

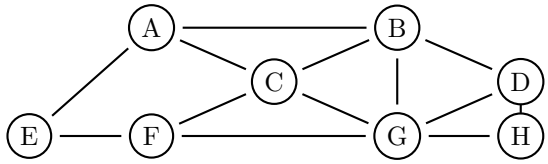
[1.0] a) Identifique, justificando, a complexidade do algoritmo descrito.

[1.5] b) Sem escrever código descreva um algoritmo mais eficiente do que o apresentado. Note que no seu algoritmo pode usar qualquer algoritmo estudado na disciplina de AED desde que identifique claramente qual o algoritmo e a sua complexidade. Identifique a complexidade global do algoritmo que descreveu.

[2.5] c) Escreva o código em C que implementa o algoritmo que descreveu na alínea anterior assumindo que a assinatura da função é a mesma que a dada acima.

Responda em folhas separadas. Não se esqueça de mudar de folha

[4.5] 16. Considere o grafo indicado em baixo à esquerda e assuma que o mesmo não é direccionado mas é ponderado, como se indica do lado direito.



D↔H	G↔H		1
A↔B,	A↔C,	B↔G	2
G↔D,	C↔G		3
B↔C,	A↔E		4
C↔F,	B↔D		5
E↔F	F↔G		6

[3.0] (a) Calcule a árvore de caminhos mais curtos (SPT) do grafo, usando o **algoritmo de Dijkstra**, tomando o vértice A como ponto de partida. Justifique detalhadamente os seus cálculos.

[1.0] (b) Trace a SPT calculada.

[0.5] (c) Identifique o caminho mais curto entre o vértice fonte e o vértice D e indique o seu custo.

Responda em folhas separadas. Não se esqueça de mudar de folha

---

[2.0] 17. Quando se utiliza a representação de um grafo através de matriz de adjacências, a maioria dos algoritmos requer um tempo $\Theta(V^2)$, mas existem algumas excepções.

Para grafos direccionados, define-se como *grau de entrada* de um vértice o número de arestas que terminam nesse vértice tendo origem nalgum outro e define-se como *grau de saída* de um vértice o número de arestas que tenham essa vértice como fonte.

Determinar se um dado grafo possui um vértice **sink** – um vértice com um grau de entrada $|V|-1$ e com grau de saída 0 – pode ser feito em tempo $\mathcal{O}(V)$, mesmo quando o grafo é representado por matriz de adjacências.

Sugestão: Todos os vértices são potenciais **sinks**. Dados dois potenciais **sinks** diferentes, u e v, se existe a aresta (u, v), u não pode ser **sink**. Se a aresta não existe, então v não pode ser **sink**. Ou seja, qualquer inspecção para um par de vértices elimina um deles como candidato.

[1.5] a) Descreva um algoritmo que, dado um grafo G, determina se existe ou não um vértice **sink** em $\mathcal{O}(V)$.

[0.5] b) Mostre por que razão o algoritmo proposto é de facto $\mathcal{O}(V)$.