DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

AED - 2016-2017 - 1º Semestre
Algoritmos e Estruturas de Dados

1^o Exame, 11 de janeiro de 2017, 11h30m 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 a seguinte tabela $id[\cdot]$ que resultou de aplicar o algoritmo da União Rápida num conjunto de ligações. Se agora forem adicionadas as seguintes sequências de ligações $(1,8)$, $(7,9)$ e $(5,4)$, quantas operações de união são realizadas?

$id[\cdot] =$

0	0	3	3	7	0	2	7	5	2

A. 0 B. 1 C. 2 D. 3

2. Considere a seguinte tabela, sobre a qual se aplica **uma vez** o algoritmo da partição, primeiro passo do algoritmo "Quicksort".

10	7	21	6	8	30	4	9

Se o elemento da partição ("pivot") for o último elemento da tabela, quantas **trocas** são efectuadas pelo algoritmo de partição **até terminar** (incluindo a troca final do "pivot")?

A. 1 B. 3 C. 4 D. 6

3. Considere a seguinte recorrência

$$C_N = 2C_{N/2} + 3N$$

Qual dos seguintes conjuntos representa a complexidade temporal associada com a recorrência dada?

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

4. Aplicou-se DFS a um grafo não direccionado, representado por matriz de adjacências, e obteve-se o seguinte caminho: 3-7-1-8-2-4-6-5.

Qual dos caminhos abaixo indicados nunca poderia ter sido obtido em DFS nesse mesmo grafo?

A. 8-4-3-7-5 B. 6-4-2-8-1 C. 1-7-3-8-2 D. 2-4-6-5-8

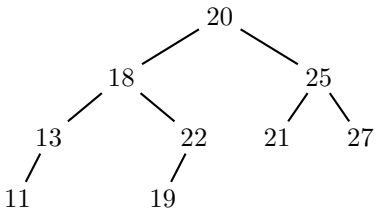
5. Considere uma tabela de dispersão de tamanho `DIM = 11`, que armazena valores inteiros. A função de dispersão é dada pelo resto da divisão do valor da chave por `DIM` (ou seja, `hash(k) = k % DIM`) e as colisões são resolvidas por **inserção em lista no início**. Considere que é inserida a seguinte série de inteiros: `35, 58, 49, 52, 71`. Depois das inserções, a tabela é percorrida de `0` a `DIM-1` e os valores armazenados nas listas são enviados sucessivamente para o terminal. Qual é o resultado obtido?

A.	35, 52, 71, 58, 49	B.	35, 58, 49, 71, 52
C.	52, 35, 71, 58, 49	D.	35, 58, 71, 49, 52

6. Considere o código à esquerda, utilizado para o varrimento da árvore binária à direita, no qual a função `visit()` imprime o valor do nó visitado.

```
void traverse(link *h, void (*visit)(link))  {
  STACKinit(max);
  STACKpush(h);

  while (!STACKempty())  {
    (*visit)(h=STACKpop());
    if (h->r != NULL) STACKpush(h->r);
    if (h->l != NULL) STACKpush(h->l);
  }
}
```



Indique qual o resultado que se obtém no final da execução da função quando chamada com a raíz da árvore.

A.	11 13 19 22 18 21 27 25 20	B.	20 18 13 11 22 19 25 21 27
C.	11 13 18 19 22 20 21 25 27	D.	20 18 25 13 22 21 27 11 19

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 *QuickSort* tem no pior caso uma complexidade $\mathcal{O}(N^2)$.

8. A complexidade de fazer N inserções ordenadas numa lista simplesmente ligada tem em média uma complexidade $\mathcal{O}(N \log N)$.

9. Dada uma tabela de N números, com N muito grande, a melhor forma de seleccionar os 10 menores números da tabela é ordenar toda a tabela e ler os 10 primeiros números.

10. Num grafo com N vértices a determinação da árvore mínima de suporte usando o algoritmo de Kruskal termina necessariamente ao fim de examinar $N - 1$ arestas.

11. A procura por um item específico numa árvore binária ordenada com N vértices tem sempre complexidade $\mathcal{O}(\log N)$.

12. Num acervo, qualquer vértice que possua dois filhos pode ter o filho mais prioritário em qualquer dos lados, esquerdo ou direito.

13. Se as arestas de um dado grafo tiverem todas o mesmo peso, a BFS permite determinar caminhos mais curtos.

14. Em tabelas de dispersão, a resolução de colisões por procura linear é melhor que a dupla dispersão por ter complexidade linear.

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. Uma sociedade de investimentos financeiros decidiu, após ter terminado o último Web Summit realizado em Lisboa, fazer uma reunião com uma das empresas que teve mais participantes registados. Para isso obteve o conjunto dos registos que foram realizados durante a conferência, identificadas pelo número de contribuinte da empresa (NIF) a que cada participante pertencia.

    Dado os fracos conhecimentos em algoritmia da sociedade financeira, contrataram-no a si para resolver este problema. No entanto, sugeriram um algoritmo que, recebendo uma tabela com os NIFs dos registos efectuados durante a conferência e o tamanho dessa tabela, revolveria o problema. Esse algoritmo foi codificado em C resultando na função `maisRegistos`.

```c
int maisRegistos(int reg[], int N)
{
  int i, j, nif, nif_max_reg, cnt, cnt_max_reg;

  nif_max_reg = reg[0];
  cnt_max_reg = 1;
  for (i=0; i<N-1; i++) {
    nif = reg[i];
    cnt = 1;
    for (j=i+1; j<N; j++) {
      if (reg[j] == nif) { cnt++; }
    }
    if (cnt > cnt_max_reg) {
      cnt_max_reg = cnt;
      nif_max_reg = nif;
    }
  }
  return nif_max_reg;
}
```

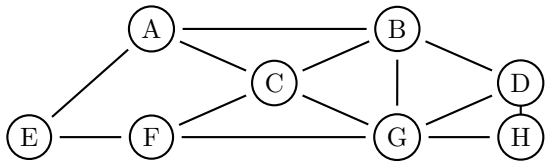
(1.0) (a) Identifique, justificando, a complexidade do algoritmo implementado pela função `maisRegistos()`. Deverá indicar a ordem de complexidade correspondente ao menor dos majorantes.

(2.0) (b) Dado os seus vastos conhecimentos de algoritmia, rapidamente percebe que o algoritmo proposto não é o mais eficiente. Sem escrever código, descreva um algoritmo mais eficiente do que o apresentado e indique, justificando, a complexidade do algoritmo que descreveu.
Note que no seu algoritmo pode usar/combinar qualquer algoritmo estudado na disciplina de AED, desde que identifique claramente o algoritmo e a sua complexidade.

(2.0) (c) Escreva o código de uma função em C que implementa o algoritmo que descreveu na alínea anterior. Não tem que escrever código para os algoritmos estudados na disciplina desde que seja claro qual a função que o realizam e os respectivos parâmetros. A função a escrever deve ter a seguinte assinatura:

```c
int maisRegEficiente(int reg[], int N);
```

[4.0] 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

(2.0) (a) Calcule a árvore mínima de suporte (MST) do grafo, usando o **algoritmo de Prim**, tomando o vértice A como ponto de partida. Justifique detalhadamente os seus cálculos.

(0.5) (b) Indique o custo da MST calculada.

(1.5) (c) O cálculo da MST de um grafo pode ser efectuado por vários algoritmos distintos.

i Diga justificando se a MST obtida com algoritmos diferentes pode ser distinta.

ii Diga justificando, se no caso de serem distintas, o custo das MST obtidas com algoritmos distintos pode igualmente ser distinto.

iii Directamente por inspeção do grafo, e sem pensar em nenhum algoritmo específico, diga justificando, quais as arestas que necessariamente teriam sempre de pertencer à MST do grafo.

[2.5] 17. Um array $v \in \mathbb{R}^{2N+1}$ diz-se ser "serpenteante", se for tal que $v[0] \leq v[1] \geq v[2] \leq v[3] \geq \cdots \leq v[2N] \geq v[2N+1]$.
Nota: a dimensão $2N+1$ apenas garante que estamos a falar de um número ímpar de elementos.

(1.0) (a) Utilizando apenas algoritmos dados na disciplina descreva um procedimento que, dado um vector $B \in \mathbb{R}^{2N+1}$, retorne um vector A contendo os mesmos elementos de B mas que seja serpenteante. Seja sucinto mas claro na descrição e não escreva qualquer código.
Nota: Será que ordenar a tabela ajuda?

(0.5) (b) Indique, justificando, qual a complexidade do procedimento que propôs, em função de N.

(0.75) (c) Assuma que existe um algoritmo `MedSelect()`, de complexidade $\mathcal{O}(N)$ que permite encontrar o elemento que corresponde à mediana de B. Utilizando este algoritmo, descreva um algoritmo para, dado B, devolver A serpenteante, de complexidade inferior ao que descreveu anteriormente.

(0.25) (d) Indique, justificando, qual a complexidade em função de N do procedimento que propôs na alínea anterior.