· U C ·

UNIVERSIDADE DE COIMBRA
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Informática

Estratégias Algorítmicas

Exame Normal – 5 de junho de 2023

Nome: _____ Nº de estudante: _____

13 pontos no total, 2 horas, sem consulta.

1. Apresente a complexidade temporal do seguinte algoritmo recursivo que retorna o índice que um número x ocupa numa lista S ou retorna -1 se não encontrar esse número na lista. Justique a sua resposta recorrendo ao Teorema Mestre. Assuma que $S = (S[1], \ldots, S[n])$ é uma lista de n números ordenados por ordem não-decrescente, que a primeira chamada deste algoritmo é $Tsearch(S, 1, n, x)$, e que as operações aritméticas demoram tempo constante. (2 pontos)

Function $Tsearch(S, \ell, r, x)$
  if $r \geq \ell$ **then**
    $i = \ell + \lfloor r - \ell)/3 \rfloor$
    $j = r - \lfloor r - \ell)/3 \rfloor$
    if $S[i] = x$ **then**
      return i
    if $S[j] = x$ **then**
      return j
    if $S[i] > x$ **then**
      return $Tsearch(S, \ell, i-1, x)$
    else if $S[j] < x$ **then**
      return $Tsearch(S, j+1, r, x)$
    else
      return $Tsearch(S, i+1, j-1, x)$
  return -1

Teorema Mestre (versão geral):
Seja $a \geq 1$, $b > 1$, $d \geq 0$.

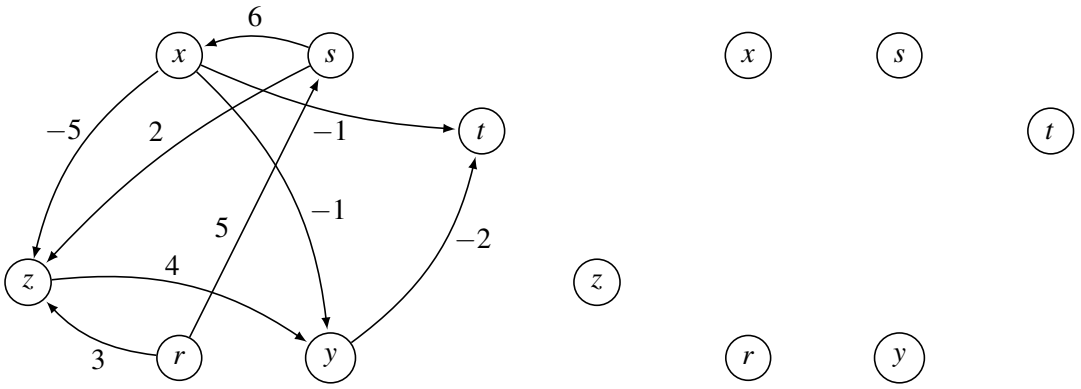
$$T(n) = \begin{cases} aT(n/b) + n^c & \text{if } n > 1 \\ d & \text{if } n = 1 \end{cases} \Rightarrow$$

$$T(n) = \begin{cases} \Theta(n^c) & \text{if } \log_b a < c \\ \Theta(n^c \log n) & \text{if } \log_b a = c \\ \Theta(n^{\log_b a}) & \text{if } \log_b a > c \end{cases}$$

2. Considere o problema de encontrar um caminho mais curto do vértice s ao vértice t num grafo $G = (V,A)$, em que V e A correspondem ao conjunto de vértices e ao conjunto de arcos, respetivamente, e em que $w(u,v)$ denota a distância entre o vértice u e o vértice v, $(u,v) \in A$. Se o grafo for acíclico, então é possível encontrar o valor do caminho mais curto entre s e t em tempo $O(|V| + |E|)$ com a seguinte abordagem.

    1. Seja $d(s) = 0$ e seja $d(v) = \infty$, para todo o vértice $v \in V \setminus \{s\}$
    2. Ordenar os vértices em V por ordenação topológica
    3. Para cada vértice $u \in V$, de acordo com a ordem topológica
        3.1 Para cada vértice v tal que $(u,v) \in A$
            3.1.1 Se $d(v) > d(u) + w(u,v)$, então $d(v) = d(u) + w(u,v)$
    4. Retorna $d(t)$

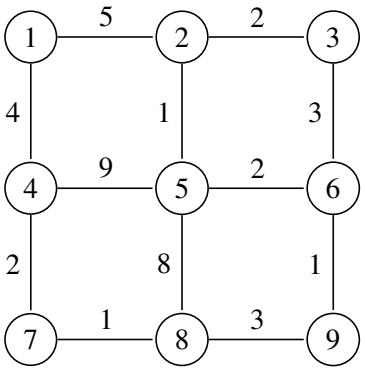
    a) Encontre um caminho mais próximo entre o vértice s e o vértice t no seguinte grafo à esquerda, com base no algoritmo descrito acima. Para o passo 2, indique, em baixo, a ordem dos vértices de acordo com a ordenação topológica. No passo 3, indique os arcos que pertencem ao caminho mais curto e os valores finais de d em cada vértice no grafo à direita. (2 pontos)



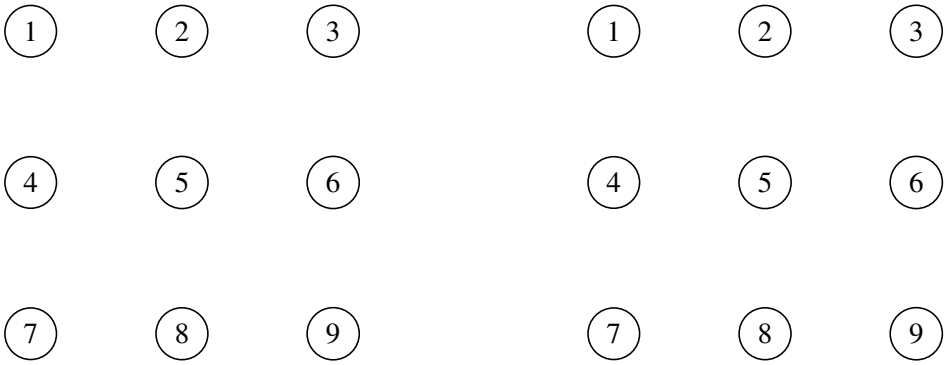
Ordem topológica dos vértices:

    b) Demonstre que o passo 3 consiste de programação dinâmica, utilizando o argumento de subestrutura ótima. (2 pontos)

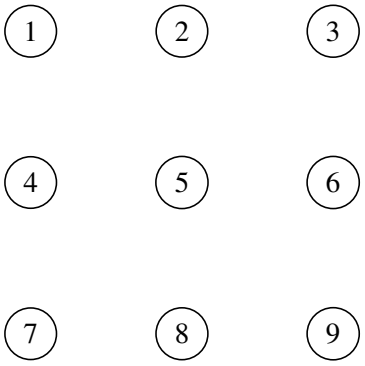
3. Considere o seguinte grafo.



a) Desenhe a árvore geradora mínima (à esqueda) e o grafo da estrutura de dados *union-find*, sem o passo de compressão de caminho (à direita), recorrendo ao algoritmo de Kruskal. Quando necessário, ligue a raiz da árvore com menor altura à raiz da árvore com maior altura e, em caso de empate, escolha, como raiz, o vértice que apresentar a etiqueta com o menor valor. (2 pontos)



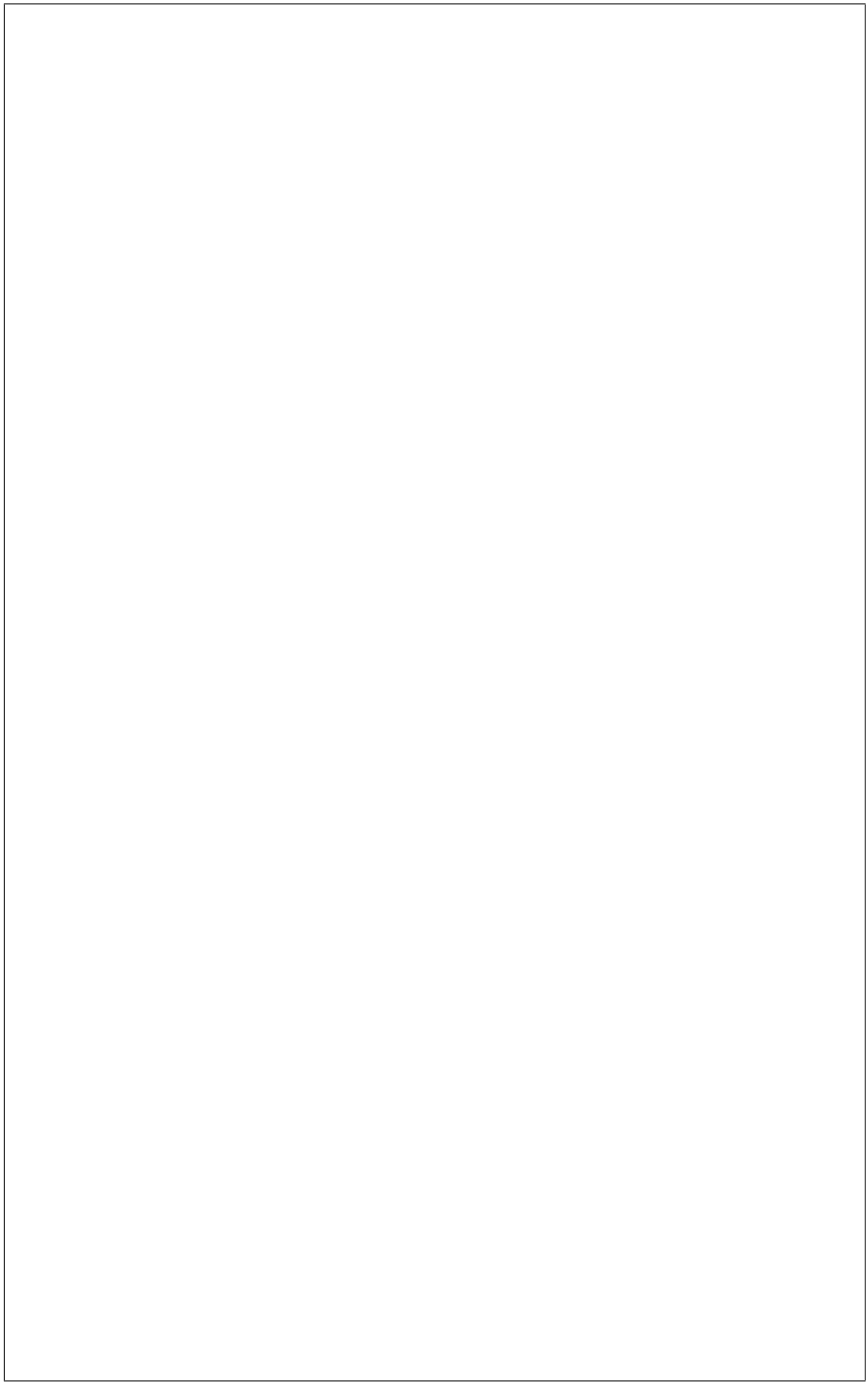
b) Qual seria a representação do grafo da estrutura de dados *union-find* se efetuasse o passo de compressão de caminho na última ligação efetuada pelo algoritmo de Kruskal? (1 ponto)

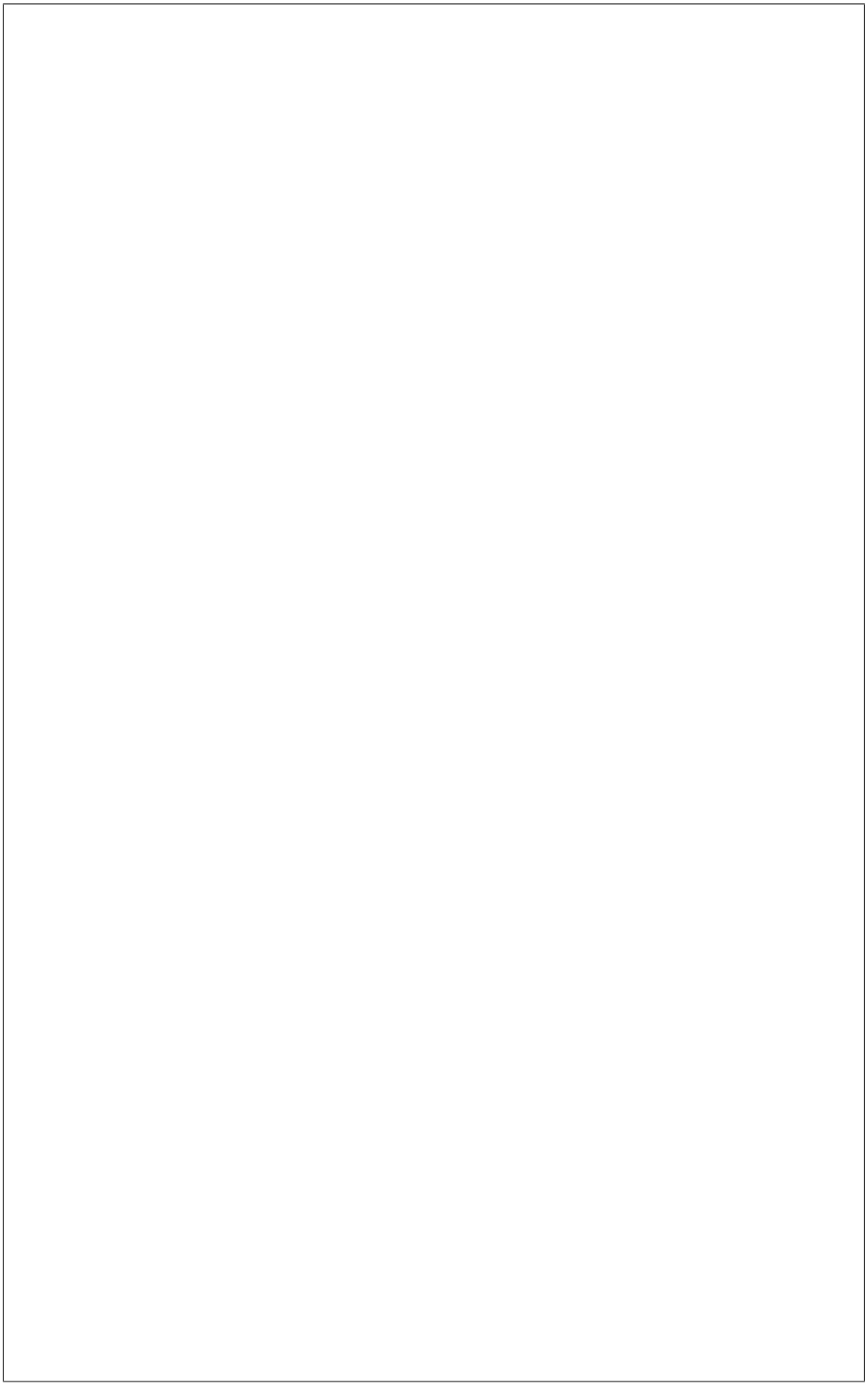


4. Um *corte* numa cadeia de caracteres consiste em dividir essa cadeia em duas sub-cadeias não-vazias. Dada uma cadeia $s = s_1 \dots s_n$, pretende-se calcular o menor número de cortes em s tal que cada sub-cadeia resultante seja um palíndromo (um palíndromo é uma palavra que pode ser lida da esquerda para a direita ou da direita para a esquerda). Por exemplo, na cadeia "ananas" o menor número de cortes é um: "anana", "s". É possível calcular o menor número de cortes pela seguinte recorrência:

$$C(s,i,j) = \begin{cases} 0 & \text{se } s_i \dots s_j \text{ é um palíndromo} \\ \min\limits_{i \le k < j} \{C(s,i,k) + 1 + C(s,k+1,j)\} & \text{caso contrário} \end{cases}$$

Com base na recorrência, apresente o pseudo-código de um algoritmo de programação dinâmica ascendente (*bottom-up*) para resolver o problema e discuta a sua complexidade computacional. Assuma a existência de uma função $Palindrome(s[i:j])$ que retorna *True* se a (sub-)cadeia $s_i, \dots, s_j$ é um palíndromo, ou *False*, caso contrário. (2 pontos)





Nome: _____ Nº de estudante: _____

5. Responda **unicamente** a uma das duas questões seguintes sobre os dois primeiros problemas práticos de Estruturas Algorítmicas. Uma versão mais compacta dos enunciados destes dois problemas está disponível nas páginas seguintes (2 pontos).

   Indique de seguida qual a questão que é considerada para avaliação (A ou B): ☐

A) **Problema A – QR Code decoder!** Considere a função *preprocessRow* seguinte que recebe uma linha do QRcode (*row*: array unidimensional de tamanho N), o tamanho da linha (N), o número de transições de cor da linha (t), o número de células pretas (p) e o número de células brancas (b). Atribua a cada célula da linha a cor preta (1) ou a cor branca (0) para o caso específico seguinte:

$$N \bmod 2 = 1 \ \wedge \ t = N - 1 \ \wedge \ |p - b| = 1$$

Explique o seu raciocínio e apresente o código da função *preprocessRow*.

Raciocínio:

Pseudo-código:

Function $preprocessRow(row, N, t, p, b)$

B) **Problema B – Expanding a Trading Portfolio** Considere a matriz bidimensional M que para cada linha $i \in \{1,\ldots,D\}$, onde D é o número de dias de acordo com o enunciado do problema, e coluna $j \in \{0,1\}$, guarda o valor ótimo que é possível obter até ao dia i (inclusive) tendo exatamente 0 ações nesse dia ($j = 0$), ou exatamente K ações nesse dia ($j = 1$).

Dada a matriz M já preenchida e os parâmetros do problema D e K, implemente o pseudo-código do método $getTradingScheme(M,D,K,S)$ que deve preencher o vetor S (indexado de 1 a D) com um "trading scheme" ótimo para a segunda tarefa do problema B.

Pseudo-código:

Function $getTradingScheme(M,D,K,S)$