



**I.d)** Considere o seguinte programa linear:

$$\begin{array}{llll} \min & -1x_1 - x_2 + x_3 & & \\ \text{s.a} & x_1 + x_2 + x_3 & \leq & 3 \\ & -2x_1 - 2x_2 - x_3 & \geq & -7 \\ & x_1 + 2x_2 + 2x_3 & \leq & 5 \\ & x_1, x_2, x_3 & \geq & 0 \end{array}$$

Indique o valor da função objectivo e o respectivo valor das variáveis básicas e não-básicas após uma única operação de pivot.

$Z$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$

II. (2,5 + 2,5 + 2,5 + 2,5 = 10 val.)

**II.a)** O Eng. Caracol foi encarregado de apresentar uma proposta para a construção de postos de primeiros socorros ao longo da autoestrada AX, que começa no quilómetro 0 e termina no quilómetro  $k$ . O Eng. Caracol dispõe de uma lista de  $n$  locais candidatos. Cada local candidato,  $1 \leq i \leq n$ , é associado à sua distância ao quilómetro 0,  $d_i$ , e ao seu custo estimado de construção,  $c_i$ . Sabendo que dois postos de primeiros socorros consecutivos não podem estar a uma distância superior a  $D$  quilómetros, o objectivo do Eng. Caracol é determinar o conjunto de locais candidatos que satisfazem a restrição do problema pelo menor custo possível.

1. Seja  $O(i)$  o custo da solução óptima para o troço da autoestrada AX entre o quilómetro 0 e o local candidato  $i$ , que atribui necessariamente um posto de primeiros socorros ao local candidato  $i$ . Defina  $O(i)$  recursivamente, completando os campos abaixo.

$$O(i) = \begin{cases} \text{ } & \text{se } i > 1 \\ \text{ } & \text{caso contrário} \end{cases}$$

2. Complete o template de código em baixo que calcula a quantidade  $O(i)$  para  $1 \leq i \leq n$  e indique a respectiva complexidade assintótica.

```

FindO( $c[1..n]$ ,  $d[1..n]$ )
    let  $O[1..n]$  be a new vector of size  $n$ 
     $O[1] = c[1]$ 
    for  $i = 2$  to  $n$  do
        
    endfor
    return  $O$ 

```

3. Explique como determinar o custo da melhor solução a partir do vector  $O$ .

**II.b)** Considere a seguinte variação do algoritmo de Rabin-Karp para emparelhamento de cadeias de caracteres sobre o alfabeto  $\Sigma = \{a, b, c\}$ , que usa a função de hash:

$$h(x_1 \dots x_n) = (\alpha(x_1) + \dots + \alpha(x_n)) \bmod 5$$

onde:  $\alpha(a) = 1$ ,  $\alpha(b) = 2$  e  $\alpha(c) = 3$ . Seja  $T = abab^2ab^3 \dots ab^{n-1}ab^n$  a string de texto a processar. Calcule o número de *hits* espúrios, em função de  $n$ , gerados ao procurar os seguintes padrões em  $T$  (deve apresentar os cálculos):

Nota:  $\sum_{i=1}^n i = \frac{n(n+1)}{2}$

1.  $P_1 = abb$

2.  $P_2 = abc$

**II.c)**

Considere o seguinte programa linear:

$$\begin{array}{llll} \max & 2x_1 & +x_2 & \\ \text{s.a} & 5x_1 & +2x_2 & \leq 60 \\ & 3x_1 & +2x_2 & \leq 40 \\ & -3x_1 & +8x_2 & \leq 40 \\ & x_1, x_2 & & \geq 0 \end{array}$$

1. Desenhe o conjunto exequível e resolva geometricamente o programa linear (indique tanto o valor máximo como as coordenadas onde esse valor é atingido).
2. Formule o programa linear dual e calcule a respectiva solução a partir da solução do programa primal (indique tanto o valor mínimo como as coordenadas onde esse valor é atingido). Nota: pode obter o valor das coordenadas resolvendo o sistema de equações do problema dual colocando a 0 as variáveis que correspondem a restrições não activas.

**II.d)** Uma matriz de incompatibilidades é uma matriz quadrada cujas células guardam valores decimais entre 0 e 1. Intuitivamente, dada uma matriz de incompatibilidades  $M$ ,  $n \times n$ , a célula  $M_{ij}$  guarda a incompatibilidade entre os índices  $i$  e  $j$ ;  $M_{ij}$  é 0 se  $i$  e  $j$  são completamente compatíveis e  $M_{ij} = 1$  se  $i$  e  $j$  são completamente incompatíveis. Dado um sub-conjunto de índices  $I \subseteq \{1, \dots, n\}$ , o nível de incompatibilidade do conjunto é dado por:  $\sum_{i,j \in I} M_{ij}$ . O problema das incompatibilidades define-se formalmente da seguinte maneira:

**Incompat** =  $\{\langle M, k, v \rangle \mid M \text{ contém um sub-conjunto de índices de tamanho } k \text{ e incompatibilidade igual ou inferior a } v\}$

1. Mostre que o problema **Incompat** está em **NP**.
2. Mostre que o problema **Incompat** é NP-difícil por redução a partir do problema **ISet**, que é sabido tratar-se de um problema NP-completo e que se define em baixo. Não é necessário provar formalmente a equivalência entre os dois problemas; é suficiente indicar a redução e a respectiva complexidade.  
*Pista:* Dado um grafo  $G$  indique como construir uma matriz de incompatibilidades cujos índices correspondem aos vértices de  $G$  tendo em conta o problema **ISet**.

*Problema ISet:* Seja  $G = (V, E)$  um grafo não dirigido; dizemos que  $V' \subseteq V$  é um conjunto de vértices independentes em  $G$  se e apenas se  $\forall u, v \in V'. (u, v) \notin E$ . O problema **ISet** define-se formalmente da seguinte maneira:

**ISet** =  $\{\langle G, k \rangle \mid G \text{ contém um conjunto de vértices independentes de tamanho } k\}$