Exercícios: Divisão e conquista e programação dinâmica

Questão 1
Temos um conjunto de números inteiros C e queremos calcular

$$C + C \mod p = \{c_1 + c_2 \mod p \mid c_1, c_2 \in C\}.$$

Exibe um algoritmo que consegue determinar $C + C \mod p$ em tempo $o(n + p^2)$, com $n = |C|$.

Exemplo: Com entrada

$$C = \{14, 15, 92, 65, 35\}; \qquad p = 19$$

o algoritmo tem que calcular

$$C + C \mod 19 = \{3, 4, 5, 9, 10, 11, 12, 13, 16\}.$$

Questão 2
A questão 1 da lista 2 pediu resolver o problema $1 \mid\mid \sum w_j C_j$. Aqui vamos resolver o problema $P2 \mid\mid \sum w_j C_j$. Nesta variante temos duas máquinas paralelas, sendo o resto do problema igual.
O problema é NP-completo. Mostra que ele é fracamente NP-completo por exibir um algoritmo que o resolve em tempo pseudo-polinomial.

Questão 3
Nesta questão vamos resolver um problema generalizado de sequenciamento de tarefas. Supõe que temos n tarefas, cada um com tempo de processamento p_j, para $j \in [n]$, e cada um com uma função de custo $c_j(t)$, $j \in [n]$, que depende do tempo de término t da tarefa. Para uma permutação π das tarefas temos tempos de término

$$C_j = \sum_{1 \leq i \leq \pi^{-1}(j)} p_{\pi_i}$$

e um custo total de

$$\sum_{j \in [n]} c_j(C_j).$$

(Para a permutação π, temos π_i é a i-ésima tarefa, logo π_j^{-1} é a posição da tarefa j.)
Dá um algoritmo que encontra uma permutação de menor custo total via programação dinâmica em tempo $O(2^n \text{poly}(n))$.