Lista 5 — MC458
Tema: Algoritmos por Indução e Algoritmos Gulosos

1. **Potenciação Rápida**
   Dados um número real x e um inteiro $n \geq 0$, queremos calcular x^n minimizando a quantidade de multiplicações executadas.

   a) Projete um algoritmo por indução em n baseado na relação $x^n = x \cdot x^{n-1}$. Prove a sua corretude por indução fraca e determine a quantidade exata de multiplicações realizadas em função de n.

   b) Considere a relação $x^n = \left(x^{\lfloor n/2 \rfloor}\right)^2$ se n for par e $x^n = x \cdot \left(x^{\lfloor n/2 \rfloor}\right)^2$ se n for ímpar. Projete o algoritmo correspondente usando indução forte e prove sua corretude. Seja $M(n)$ o número de multiplicações com $M(1) = 0$, mostre que $M(n) \leq M(\lfloor n/2 \rfloor) + 2$ e conclua por indução forte que $M(n) \leq 2\lfloor \log_2 n \rfloor$ para todo $n \geq 1$.

2. **Parênteses Balanceados**
   Uma sequência de parênteses `(` e `)` é dita balanceada se, lida da esquerda para a direita, a quantidade de `(` nunca fica menor do que a quantidade de `)` em nenhum de seus prefixos, e ambas as quantidades são iguais ao final da sequência.

   a) Mostre que a hipótese de indução fraca *"sabemos dizer se uma sequência de $n-1$ símbolos é balanceada"* é insuficiente para projetar o algoritmo. Utilize as sequências não balanceadas `(` e `((` para demonstrar o que ocorre ao acrescentar o símbolo `)` ao final de cada uma.

   b) Reforce a hipótese de indução para: *"sabemos calcular o saldo (quantidade de `(` menos a quantidade de `)`) e identificar se algum prefixo possuiu saldo negativo"*. Projete o algoritmo baseado nessa hipótese reforçada, prove sua corretude por indução no tamanho da sequência e demonstre que ele executa em tempo $O(n)$.

3. **Cobertura de Pontos por Antenas de Raio Fixo**
   Uma empresa de telecomunicações precisa instalar antenas ao longo de uma rodovia retilínea para fornecer sinal a n vilas localizadas nas posições $x_1 < x_2 < \cdots < x_n$. Cada antena instalada em uma posição p possui um raio de alcance R, cobrindo o intervalo $[p - R, p + R]$. O objetivo é cobrir todas as n vilas utilizando o menor número possível de antenas.

   a) Um projetista sugeriu a seguinte estratégia gulosa: *"Encontre a primeira vila não coberta x_i e posicione a antena exatamente sobre ela ($p = x_i$)"*. Apresente um contraexemplo numérico simples (indicando posições x_i e o raio R) demonstrando que essa estratégia pode instalar mais antenas do que o necessário.

   b) Formule a estratégia gulosa correta para este problema. Prove a corretude do seu algoritmo (por exemplo, utilizando o argumento de que *"o guloso não fica para trás"*) e determine sua complexidade de tempo em função do número de vilas n, assumindo que a entrada já se encontra ordenada.

4. **Minimização do Atraso Máximo em Tarefas)**
Um servidor precisa processar n tarefas consecutivas. Cada tarefa i requer $p_i > 0$ unidades de tempo de processamento contínuo e possui um prazo limite d_i. Todas as tarefas encontram-se disponíveis no instante $t = 0$. Se a tarefa i é concluída no instante C_i, seu atraso é dado por $A_i = \max(0, C_i - d_i)$. O objetivo é determinar uma ordem de execução que minimize o atraso máximo $A_{\max} = \max_{1 \le i \le n} A_i$.

   a) Apresente contraexemplos pequenos mostrando que nenhuma das duas heurísticas intuitivas a seguir garante o atraso máximo mínimo:
      i) *Menor Duração Primeiro:* Ordenar as tarefas por p_i crescente.
      ii) *Menor Folga Primeiro:* Ordenar as tarefas por $d_i - p_i$ crescente.

   b) Considere o algoritmo **EDD** (*Earliest Deadline First*), que ordena as tarefas em ordem crescente de prazo ($d_1 \le d_2 \le \cdots \le d_n$). Chamamos de **inversão** qualquer par de tarefas consecutivas na sequência em que uma tarefa com prazo maior é executada imediatamente antes de uma tarefa com prazo menor ($d_i > d_{i+1}$).
      i) Mostre que, se uma sequência possui uma inversão entre as posições i e $i + 1$, trocar a posição dessas duas tarefas nunca aumenta o atraso máximo do par.
      ii) Conclua por que qualquer ordenação pode ser transformada na ordem EDD eliminando inversões uma a uma sem piorar o atraso máximo do sistema, provando a otimidade do algoritmo EDD.

5. **O Problema do Motorista e Postos de Combustível**
Um motorista precisa viajar do ponto A ao ponto B ao longo de uma estrada retilínea de comprimento D. O tanque de combustível do veículo, quando cheio, permite percorrer no máximo C quilômetros. Existem n postos de combustível localizados nas distâncias $x_1 < x_2 < \cdots < x_n$ a partir do ponto A (onde $x_0 = 0$ é o ponto A e $x_{n+1} = D$ é o ponto B). Assume-se que $x_{i+1} - x_i \le C$ para todo i. O objetivo é ir de A a B efetuando o menor número total de paradas para reabastecimento.

   a) Considere a seguinte regra de parada: *"Se ao chegar ao posto x_i o combustível restante no tanque for menor que $C/2$, pare e reabasteça; caso contrário, siga sem parar"*. Apresente um contraexemplo no qual essa regra comete mais paradas do que o necessário ou falha em chegar ao destino B.

   b) Formule a estratégia gulosa correta (*"dirija o máximo possível com o combustível atual e pare no último posto alcançável dentro da autonomia"*), apresente o pseudocódigo do algoritmo e prove por indução que ele minimiza o número de paradas.

6. **Organização Ótima de Arquivos em Fita Magnética**
Um sistema grava n arquivos de tamanhos $L_1, L_2, \ldots, L_n$ sequencialmente em uma fita magnética. Para ler o k-ésimo arquivo armazenado na fita, o leitor precisa percorrer todos os $k - 1$ arquivos gravados antes dele. Assim, o tempo de acesso ao k-ésimo arquivo é a soma dos tamanhos do 1º até o k-ésimo arquivo. Queremos determinar a ordem de gravação que minimize a soma total dos tempos de acesso $T_{\text{total}} = \sum_{k=1}^{n} T_k$.

   a) Calcule o tempo total de acesso T_{total} para 3 arquivos de tamanhos $L = [2, 5, 10]$ sob duas estratégias distintas:
      i) Ordem decrescente de tamanho: $[10, 5, 2]$
      ii) Ordem crescente de tamanho: $[2, 5, 10]$

b) Prove a otimidade da ordem crescente por análise de contribuição de pesos:

i) Mostre que, para qualquer ordenação $\pi(1), \pi(2), \ldots, \pi(n)$, o tempo total de acesso pode ser escrito como a soma ponderada:

$$T_{\text{total}} = n \cdot L_{\pi(1)} + (n-1) \cdot L_{\pi(2)} + \cdots + 1 \cdot L_{\pi(n)}$$

ii) Utilizando o princípio de que a soma de produtos é minimizada associando os maiores pesos aos menores valores, explique por que ordenar os arquivos em ordem crescente de tamanho ($L_{\pi(1)} \leq L_{\pi(2)} \leq \cdots \leq L_{\pi(n)}$) garante o tempo total mínimo.