Universidade Federal do Ceará - Campus Crateús
Disciplina: Projeto e Análise de Algoritmos
Professor: Rennan Dantas

Nome do Aluno(a):
Matrícula:

Lista de exercícios para o teste 1

Questão 01 A multiplicação à Russa consiste em:

a. Escrever os números A e B, que se deseja multiplicar na parte superior das colunas.

b. Dividir A por 2, sucessivamente, ignorando o resto até chegar à unidade, escrever os resultados da coluna A

c. Multiplicar B por 2 tantas vezes quantas se haja dividido A por 2, escrever os resultados sucessivos na coluna B

d. Somar todos os números da coluna B que estejam ao lado de um número ímpar da coluna A

Exemplo : 27×82

A	B	Parcelas
27	82	82
13	164	164
6	328	-
3	656	656
1	1312	1312
	Soma:	**2214**

Escrever uma recursão que permita fazer a multiplicação à Russa de 2 entradas. Explique porque a sua recursão funciona.

Questão 02 (POSCOMP - Adaptado) Considere a relação de recorrência a seguir:

$$T(n+1) = \begin{cases} d, & \text{se n=0} \\ n \times T(n), & \text{se n>0} \end{cases}$$

Considere d um valor constante. Com base nessa relação de recorrência, determine, via Árvore de recursão/Expansão telescópica, o valor fechado dessa recursão.

Questão 03 Reza a lenda que no começo dos tempos, Deus criou a Torre de Brahma que contém três pinos de diamante e colocou no primeiro 64 discos de ouro maciço. Deus então chamou os seus sacerdotes e ordenou-lhes que transferissem todos os discos para o terceiro pino, segundo as regras abaixo. Os sacerdotes então obedeceram e começaram o seu trabalho, dia e noite. Quando eles terminarem, a Torre de Brahma irá ruir e o mundo acabará.

Os discos possuem diâmetros diferentes (não existem dois discos com o mesmo diâmetro) e todos possuem um furo em seu centro tal que podem ser encaixados em qualquer um dos pinos. Inicialmente os discos formam uma torre onde todos são colocados em um dos pinos e em ordem decrescente de tamanho (no topo está o disco de maior diâmetro e na base o de maior diâmetro). Devemos transferir toda a torre para um dos outros pinos de modo que cada movimento é feito somente com um disco, nunca havendo um disco maior sobre um disco menor.

Qual o menor número de movimentos necessários para resolver uma torre de Hanói com n discos? Justifique sua resposta.

Questão 04 Prove as seguintes a afirmações sobre notação assintótica:

a) $n^3/100 - 25n^2 - 100n + 7$ é $\Omega(n^2)$ e $\Theta(n^3)$

b) $77n^3 - 13n^2 + 29n - 5$ é $\mathrm{O}(n^4)$ e $\Omega(n^3)$

c) $34n\ log_7\ n^2 + 13n$ é $\Omega(n)$ e $\mathrm{O}(n^2)$

Questão 05 (POSCOMP) Considere dois algoritmos A_1 e A_2, cujas funções de custo são, respectivamente, T_1(n)=$n^2 - n + 1$ e T_2(n)=6n log_2 n + 2n. Para simplificar a análise, assuma que n>0 é sempre uma potência de 2. Com relação ao enunciado, assinale a alternativa correta.

a) Como $T_1(n) = \Theta(n^2)$ e T_2(n)=Θ(n log n), então A_2 é sempre mais eficiente que A_1.
b) O limite superior de T_1(n)=O(n^3) é correto e assintoticamente restrito.
c) O limite inferior T_2(n)=Ω(n^3) é correto e assintoticamente restrito.
d) T_1 e T_2 são assintoticamente equivalentes.
e) A_1 é mais eficiente que A_2, para n suficientemente pequeno.

Justifique sua resposta.

OBS: Assintoticamente restrito se refere a um limite "firme". Em outras palavras, ser Θ.

Questão 06 Resolva as seguintes equações de recorrência segundo o método da árvore de recursão:

a) $T(n) = 2T(n-2) + 1$

b) $T(n) = 5T(n/2) + n^2$

c) $T(n) = T(0.99n) + 7$

d) $T(n) = T(\sqrt{n}) + log_2\ n$

e) $T(n) = 2T(\sqrt{n}) + log_2\ n$

f) $T(n) = 2T(n/5) + 3T(n/6) + n$

Questão 07 (POSCOMP) Considere o algoritmo a seguir:

PROC (n)

se (n <= 1) então

retorna 1+n;

senão

retorna PROC(n/2) + PROC(n/2);

fim se

Assinale a alternativa que indica corretamente quantas comparações são feitas para uma entrada n > 0, onde n é um número natural.

a) n
b) log n + 1
c) n log n + 1
d) n^2 + n - 1
e) 2n - 1

Justifique sua resposta.

Questão 08 Faça um algoritmo de tempo Θ(n log n) para resolver o seguinte problema: dado um vetor com n números inteiros positivos e um outro número inteiro positivo x, determine se existem ou não dois elementos cuja soma é igual a x.

Questão 09 Um vetor bitônico é um vetor de números que consiste de uma primeira parte ordenada de forma crescente seguida de uma segunda parte ordenada em forma decrescente. Mais precisamente, um vetor A[1..n] é bitônico se e somente se existe um índice i, $1 \leq i \leq n$, tal que A[1..i] é estritamente crescente e A[i..n] é estritamente decrescente.
Por exemplo, o vetor [2,5,8,7] é bitônico com i=3.
Forneça um algoritmo de divisão e conquista que encontre o maior elemento no vetor bitônico (você pode assumir que todos os elementos são distintos) em tempo O(log n).

Atenção: Explique resumidamente como o seu algoritmo funciona E prove a complexidade dele.
Obs: Você não deve usar memória auxiliar como vetores.

Questão 10 Sejam X[1..n] e Y[1..n] dois vetores ordenados. Escreva um algoritmo Θ(log n) para encontrar a mediana de todos os 2n elementos nos vetores X e Y. Prove esta complexidade.

Questão 11 Diz-se que um vetor A[1..n] possui um elemento majoritário se mais do que a metade de seus elementos são iguais. Dado um vetor, a tarefa é projetar um algoritmo eficiente para dizer se o vetor possui um elemento majoritário e, se afirmativo, encontrar esse elemento. Os elementos do vetor não são necessariamente de algum domínio ordenado como os inteiros e, portanto, não podem existir comparações do tipo "A[i]>A[j]?" (os elementos desse vetor não são necessariamente inteiros ou caracteres; podem ser objetos quaisquer, como frutas ou arquivos executáveis). Entretanto, você pode responder questões da forma "A[i]=A[j]?" em tempo constante. Utilizando divisão e conquista, mostre como resolver este problema em tempo O(n log n). Você não deve utilizar memória adicional (outro vetor, alguma matriz). Justifique porque o seu algoritmo funciona.

Dica: Divida o vetor A em dois vetores A_1 e A_2, cada um com metade do tamanho do vetor A. Conhecer o elemento majoritário de A_1 e de A_2 te ajuda a encontrar o elemento majoritário de A?

Questão 12 Seja X[1..n] um vetor de inteiros. Dados i < j em $\{1..n\}$, dizemos que (i,j) é uma inversão de X se X[i] > X[j]. Escreva um algoritmo Θ(n log n) que devolva o número de inversões em um vetor X.

Questão 13 Em sala de aula, vimos sem detalhes um algoritmo recursivo de divisão e conquista para multiplicar duas matrizes quadradas, dividindo cada matriz em 4 submatrizes quadradas e realizando 8 chamadas recursivas (não estamos falando do algoritmo de Strassen que realiza 7 chamadas recursivas, mas do mais simples visto antes dele). Descreva como seria um algoritmo que divida cada matriz em 9 submatrizes quadradas. Calcule a complexidade de tempo em notação assintótica.

Questão 14 Seja X[1..n] um vetor de números reais. Dizemos que X tem um elemento popular x se mais de um terço de seus elementos são iguais a x. Escreva um algoritmo de tempo linear Θ(n) que diz se X possui ou não um elemento popular. Caso sim, devolva o seu valor. Dica: Use o algoritmo de Seleção do k-ésimo mínimo de tempo linear no pior caso.

Questão 15 Você recebe um vetor de n elementos e percebe que alguns dos elementos deste vetor possuem duplicatas; isto é, eles aparecem mais de uma vez no vetor. Mostre um algoritmo que remove, utilizando divisão e conquista, todas as duplicatas desse vetor em tempo O(n log n).

Dica 1: Você não precisa se preocupar com a operação de remover um elemento de um vetor. Você pode simplesmente usar uma operação "Remover(x)" que remove o elemento x.

Dica 2: Você pode usar os algoritmos clássicos (os de ordenação e os seus auxiliares) apenas chamando-os. Exemplos: Quicksort(A) ou Mergesort(A), ordenam o vetor A; Partition(A), particiona o vetor A; Intercala(A,B), intercala os elementos de A e B.

Dica 3: Um algoritmo de ordenação deve ajudar você, haja vista que eles comparam muitos elementos...você pode remover aqueles que uma comparação aponta como iguais por exemplo.