UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO
http://www.icmc.usp.br

SCC5900 - Projeto de Algoritmos
Lista de Exercícios

Questão 1 Considere o algoritmo baseado na definição da soma de duas matrizes de dimensões n x n.

a) Qual é a operação básica do algoritmo?
b) Quantas vezes essa operação é realizada em função da ordem n das matrizes?
c) Quantas vezes essa operação é realizada em função do número total de elementos das matrizes de entrada?

Questão 2 Responda as questões do exercício anterior considerando o algoritmo baseado na definição da multiplicação de matrizes.

Questão 3 Considere uma modificação da busca sequencial que varre uma lista e retorna o número de ocorrências de uma dada chave nessa lista. Sua eficiência computacional será diferente da eficiência da busca clássica?

Questão 4 Para cada um dos pares de funções a seguir, identifique a função com maior ordem de crescimento e os pares tais que ambas as funções possuem mesma ordem.

a) $n(n+1)$ e $2000n^2$
b) $100n^2$ e $0,01n^3$
c) $log_2(n)$ e $ln(n)$
d) $log_2^2(n)$ e $log_2(n^2)$
e) 2^{n-1} e 2^n
f) $(n-1)!$ e $n!$

Questão 5 Segundo uma lenda muito conhecida, o jogo de xadrez foi inventado séculos atrás na Índia por um sábio chamado Sashi. Quando ele levou sua invenção ao rei, o rei gostou tanto do jogo que permitiu ao inventor escolher qualquer recompensa que desejasse. Sashi pediu uma porção de grãos a ser calculada de acordo com o seguinte algoritmo: "um único grão de trigo deveria ser colocado em uma primeira casa do tabuleiro, dois em uma segunda, quatro em uma terceira, oito em uma quarta casa e assim por diante até completar as 64 casas do tabuleiro". Qual seria a quantidade de trigo resultante calculada por esse algoritmo?

Questão 6 Use as definições informais de O, Θ e Ω para determinar se as afirmativas a seguir são verdadeiras.

a) $n(n+1)/2 \in O(n^3)$
b) $n(n+1)/2 \in O(n^2)$
c) $n(n+1)/2 \in \Theta(n^3)$
d) $n(n+1)/2 \in \Omega(n)$

Questão 7 Para cada uma das funções a seguir, indique a classe $\Theta(g(n))$ à qual a função pertence. (Use o menor $g(n)$ possível nas suas respostas.) Prove suas afirmativas.

a) $(n^2+1)^{10}$
b) $\sqrt{10n^2+7n+3}$
c) $2nlg(n+2)^2+(n+2)^2lg(\frac{n}{2})$
d) $2^{n+1}+3^{n-1}$
e) $\lfloor log_2(n) \rfloor$

Questão 8 Ordene as funções a seguir de acordo com suas ordens de crescimento (da menor para a maior ordem de crescimento):

$$(n-2)!,\ 5lg(n+100)^{10},\ 2^{2n},\ 0,001n^4+3n^3+1,\ ln^2(n),\ \sqrt[3]{n},\ 3^n$$

Questão 9 Você está diante de uma parede que se estende infinitamente em ambas as direções. Há uma porta nessa parede, mas você não sabe quão longe ela está ou se ela se encontra à sua esquerda ou à sua direita. Você apenas pode ver a porta quando está exatamente em frente a ela. Determine um algoritmo que permita a você encontrar a porta caminhando no máximo $O(n)$ passos, sendo que n é o número (desconhecido) de passos que separa sua posição inicial da posição da porta.[1]

Questão 10 Considere o algoritmo a seguir.

ALGORITMO *Desconhecido*(n)
    //Entrada: Um inteiro não-negativo n
    $S \leftarrow 0$
    para $i \leftarrow 1$ **até** n **faça**
        $S \leftarrow S + i * i$
    retorne S

Responda:

a) O que esse algoritmo calcula?
b) Qual sua operação básica?
c) Quantas vezes essa operação básica é executada?
d) A qual a classe de eficiência pertence esse algoritmo?
e) Sugira uma melhoria ou um algoritmo mais eficiente e indique sua classe de eficiência. Caso não seja possível, demonstre.

Questão 11 Considere o algoritmo a seguir.

ALGORITMO *Segredo*$(A[0..n-1])$
    //Entrada: Um vetor $A[0..n-1]$ de números reais
    $min \leftarrow A[0]$
    $max \leftarrow A[0]$
    para $i \leftarrow 1$ **até** $n-1$ **faça**
        se $A[i] < min$
            $min \leftarrow A[i]$
        se $A[i] > max$
            $max \leftarrow A[i]$
    retorne $max - min$

Responda as questões do exercício anterior para o algoritmo *Segredo*.

[1]Parberry, I. *Problems on Algorithms*. Prentice-Hall, Englewood Clifs, NJ, 1995 – nº 652.

Questão 12 Considere o algoritmo a seguir.

ALGORITMO *Enigma*($A[0..n-1, 0..n-1]$)
//Entrada: Uma matriz $A[0..n-1, 0..n-1]$ de números reais
para $i \leftarrow 0$ **até** $n-2$ **faça**
  para $j \leftarrow i+1$ **até** $n-1$ **faça**
    se $A[i,j] \neq A[j,i]$
      retorne falso
retorne verdadeiro

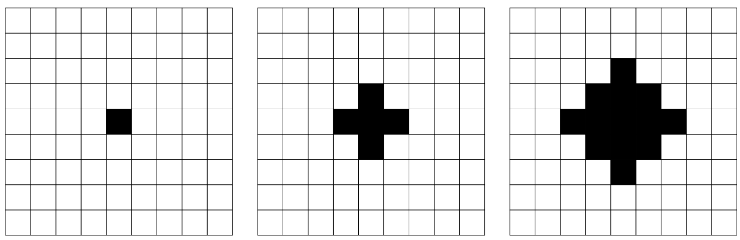
Responda as questões dos exercícios anteriores para o algoritmo *Enigma*.

Questão 13 Considere a seguinte versão de um importante algoritmo.

ALGORITMO *GE*($A[0..n-1, 0..n]$)
//Entrada: Uma matriz de dimensões n x n $A[0..n-1, 0..n]$ de números reais
para $i \leftarrow 0$ **até** $n-2$ **faça**
  para $j \leftarrow i+1$ **até** $n-1$ **faça**
    para $k \leftarrow i$ **até** n **faça**
      $A[j,k] \leftarrow A[j,k] - A[i,k] * A[j,i]/A[i,i]$

a) Encontre a classe de eficiência desse algoritmo.
b) Qual característica obviamente ineficiente esse algoritmo contém e como ela pode ser eliminada para tornar o algoritmo mais rápido?

Questão 14 Quantos quadrados unitários são produzidos pelo algoritmo que começa com um único quadrado e a cada uma de suas n iterações adiciona quadrados ao lado de cada quadrado em sua fronteira?[2] (No jargão de teoria dos autômatos celulares, a resposta é o número de células na vizinhança de von Neumann de alcance n.) Os resultados para $n = 0$, 1 e 2 são ilustrados na figura abaixo.



Questão 15 Resolva as relações de recorrência a seguir.

a) $x(n) = x(n-1) + 5$, para $n > 1$, $x(1) = 0$
b) $x(n) = 3x(n-1)$, para $n > 1$, $x(1) = 4$
c) $x(n) = x(n-1) + n$, para $n > 0$, $x(0) = 0$
d) $x(n) = x(n/2) + n$, para $n > 1$, $x(1) = 1$ (resolva para $n = 2^k$)
e) $x(n) = x(n/3) + 1$, para $n > 1$, $x(1) = 1$ (resolva para $n = 3^k$)

Questão 16 Estabeleça e resolva uma relação de recorrência para o número de chamadas feitas por $F(n)$, o algoritmo recursivo para calcular $n!$.

[2] Gardiner, A. *Mathematical and Logic Puzzles.* Dover Publications, New York, 1994. – página 88.

Questão 17 Considere o algoritmo recursivo a seguir para computar a soma dos primeiros n cubos inteiros $S(n) = 1^3 + 2^3 + \ldots + n^3$.

ALGORITMO $S(n)$
//Entrada: Um inteiro positivo n
//Saída: A soma dos primeiros n cubos
se $n = 1$
  retorne 1
senão
  retorne $S(n-1) + n * n * n$

a) Estabeleça e resolva uma relação de recorrência para o número de vezes que a operação básica do algoritmo é realizada.

b) Compare esse algoritmo com a versão não-recursiva mais simples do algoritmo para calcular essa mesma soma.

Questão 18 Considere o algoritmo recursivo a seguir.

ALGORITMO *Min1*$(A[0..n-1])$
//Entrada: Um vetor $A[0..n-1]$ de números reais
se $n = 1$
  retorne $A[0]$
senão
  $temp \leftarrow Min1(A[0..n-2])$
  se $temp \leqslant A[n-1]$
    retorne $temp$
  senão
    retorne $A[n-1]$

a) O que esse algoritmo computa?

b) Estabeleça uma relação de recorrência para a o número de vezes que o algoritmo realiza sua operação básica e solucione essa relação.

Questão 19 Considere outro algoritmo para solucionar o problema do exercício anterior, o qual divide o vetor recursivamente em duas metades, a primeira chamada do algoritmo sendo $Min2(A[0..n-1])$, onde

ALGORITMO *Min2*$(A[l..r])$
//Entrada: Um vetor $A[l..r]$ de números reais
se $l = r$
  retorne $A[l]$
senão
  $temp1 \leftarrow Min2(A[l..\lfloor (l+r)/2 \rfloor])$
  $temp2 \leftarrow Min2(A[\lfloor (l+r)/2 \rfloor + 1..r])$
  se $temp1 \leqslant temp2$
    retorne $temp1$
  senão
    retorne $temp2$

a) Estabeleça uma relação de recorrência para a o número de vezes que o algoritmo realiza sua operação básica e solucione essa relação.

b) Qual dos algoritmos, $Min1$ ou $Min2$, é mais rápido? Você seria capaz de sugerir um algoritmo que resolvesse o problema mais eficientemente que ambos?

Questão 20 Considere os questionamentos.

a) Desenvolva um algoritmo recursivo para computar 2^n para qualquer inteiro não-negativo n que odebeça a fórmula $2^n = 2^{n-1} + 2^{n-1}$.

b) Estabeleça uma relação de recorrência para o número de operações de adição realizadas pelo algoritmo e solucione essa relação.

c) Desenhe uma árvore representando as chamadas recursivas do algoritmo e conte o número de chamadas realizadas pelo algoritmo.

d) Avalie: esse é um bom algoritmo para resolver o problema?

Questão 21 Considere o algoritmo recursivo a seguir.

ALGORITMO $Q(n)$
  //Entrada: Um inteiro positivo n
  se $n = 1$
    retorne 1
  senão
    retorne $Q(n-1) + 2 * n - 1$

a) Estabeleça uma relação de recorrência para os valores dessa função e solucione essa relação para determinar o que esse algoritmo computa.

b) Estabeleça uma relação de recorrência para o número de operações de multiplicação feitas pelo algoritmo e solucione essa relação.

c) Estabeleça uma relação de recorrência para o número de operações de adição/subtração feitas pelo algoritmo e solucione essa relação.

Questão 22 Considere o conhecido algoritmo de ordenação a seguir com um contador embutido para contar o número de comparações de chaves.

ALGORITMO $Ordena(A[0..n-1])$
  //Entrada: Um vetor $A[0..n-1]$ de elementos "ordenáveis"
  //Saída: O número de comparações efetuadas
  $contador \leftarrow 0$
  para $i \leftarrow 1$ **até** $n-1$ **faça**
    $v \leftarrow A[i]$
    $j \leftarrow i - 1$
    enquanto $j \geqslant 0$ **e** $A[j] > v$ **faça**
      $contador \leftarrow contador + 1$
      $A[j+1] \leftarrow A[j]$
      $j \leftarrow j - 1$
    $A[j+1] \leftarrow v$
  retorne $contador$

O incremento do contador está sendo feito no local correto? Se afirmativo, demonstre; caso contrário, faça uma correção apropriada.

Questão 23 Considere novamente o agoritmo do exercício anterior ou, caso tenha sido necessário, a sua versão corrigida do algoritmo com o incremento do contador em seu lugar correto.

a) Execute o algoritmo sobre vinte vetores com valores aleatórios e tamanhos 1.000, 1.500, 2.000, …, 9.500.

b) Analise os dados obtidos para formular uma hipótese sobre a eficiência do algoritmo no caso médio.

c) Estime o número de comparações de chaves esperado para um vetor com valores aleatórios de tamanho 10.000.

Questão 24 Refaça os passos do exercício anterior analisando o tempo de execução do programa em milissegundos.

Questão 25 Faça uma análise do tempo de execução dos três algoritmos para computar o máximo divisor comum entre dois número m e n, $gcd(m, n)$, apresentados na Seção 1.1 do livro-texto da disciplina (Levitin).

Questão 26 A variância amostral de n medidas $x_1, x_2, \ldots, x_n$ pode ser calculada como

$$\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1} \quad \text{, sendo que} \quad \overline{x} = \frac{\Sigma_{i=1}^{n} x_i}{n} \tag{1}$$

ou

$$\frac{\Sigma_{i=1}^{n} x_i^2 - \frac{(\Sigma_{i=1}^{n} x_i)^2}{n}}{n-1}$$

Calcule e compare o número de operações de divisões, multiplicações e adições/subtrações (adições e subtrações são tipicamente enumeradas como um grupo) necessárias para computar a variância de acordo com cada uma das fórmulas.