

LISTA DE EXERCÍCIOS 1

As questões, a seguir, tratam do projeto de algoritmos. CLSR é a referência do exercício no livro-texto.

Questão 1 [CLSR, Ex 2.2-4, pg 27]

Como podemos modificar quase que qualquer algoritmo para ter um bom tempo de execução para o melhor caso?

Questão 2 [CLSR, Ex 2.3-4, pg 37]

O algoritmo de Ordenação por Inserção pode ser expresso como um procedimento recursivo da seguinte forma: para ordenar $A[1..n]$, ordena-se recursivamente $A[1..n-1]$ e então insere-se $A[n]$ no vetor ordenado $A[1..n-1]$. Escreva uma equação de recorrência para o tempo de execução dessa versão recursiva, resolva essa equação e indique o crescimento da pilha de recursão.

Questão 3 [CLSR, Ex 2.3-6, pg 37]

Observe que o laço **while** das linhas 5–7 do procedimento INSERTION-SORT (apresentado na seção 2.1 do livro CLSR, pg 24 e mostrado abaixo) usa uma pesquisa linear para percorrer de trás para frente o vetor ordenado $A[1..j-1]$. Pode-se usar a pesquisa binária para melhorar o tempo de execução no pior caso do INSERTION-SORT para $\Theta(n \log n)$?

INSERTION-SORT(A)

```
1  for  $j \leftarrow 2$  to  $\text{length}[A]$ 
2      do  $\text{key} \leftarrow A[j]$ 
3          ▷ Insert  $A[j]$  into the sorted sequence  $A[1..j-1]$ .
4           $i \leftarrow j-1$ 
5          while  $i > 0$  and  $A[i] > \text{key}$ 
6              do  $A[i+1] \leftarrow A[i]$ 
7                   $i \leftarrow i-1$ 
8           $A[i+1] \leftarrow \text{key}$ 
```

Questão 4 [CLSR, Ex 2.3-6, pg 37]

Descreva um algoritmo com complexidade de tempo $\Theta(n \log n)$ que, dado um conjunto S de n inteiros e um outro inteiro x , determina se existe ou não dois elementos de S cuja soma é exatamente x .

Questão 5 [CLSR, Problema 2-1, pg 37]

(*Ordenação por Inserção em pequenos vetores no Mergesort.*) Sabe-se que o algoritmo Mergesort executa no pior caso em tempo $\Theta(n \log n)$ e o algoritmo de Ordenação por Inserção no pior caso em tempo $\Theta(n^2)$. No entanto, os fatores constantes do Inserção o tornam mais rápido para um pequeno valor de n . Assim, faz sentido usar o algoritmo de Ordenação por Inserção quando os sub-problemas tornam-se suficientemente pequenos. Considere a seguinte modificação no Mergesort: $\frac{n}{k}$ sub-listas de comprimento k são ordenadas usando o algoritmo de Ordenação por Inserção e então combinadas (*merged*) usando o mecanismo do Mergesort, sendo k o valor a ser determinado.

- (a) Mostre que as $\frac{n}{k}$ sub-listas, cada uma de comprimento k , podem ser ordenadas pelo algoritmo de Ordenação por Inserção no pior caso em tempo $\Theta(nk)$.

- (b) Mostre que as sub-listas podem ser combinadas (*merged*) no pior caso em tempo $\Theta(n \log \frac{n}{k})$.
- (c) Dado que o algoritmo modificado executa no pior caso em tempo $\Theta(nk + n \log \frac{n}{k})$, qual é o maior valor assintótico (usando notação Θ) de k como uma função de n para o qual o algoritmo modificado tem o mesmo tempo de execução assintótico do Mergesort padrão?
- (d) Na prática, como o valor de k seria escolhido?

Questão 6 [CLSR, Problema 2-4, pg 39]

(*Inversões.*) Seja $[A \dots n]$ um vetor com n números distintos. Se $i < j$ e $A[i] > A[j]$, então o par (i, j) é chamado uma inversão de A .

- (a) Liste as cinco inversões do vetor $\langle 2, 3, 8, 6, 1 \rangle$.
- (b) Que vetor com elementos do conjunto $\{1, 2, \dots, n\}$ tem o maior número de inversões? Quantas inversões existem?
- (c) Qual é a relação entre o tempo de execução do algoritmo de Ordenação por Inserção e o número de inversões do vetor de entrada? Justifique sua resposta.
- (d) Apresente um algoritmo que determina o número de inversões em qualquer permutação de n elementos no pior caso em tempo $\Theta(n \log n)$. (Dica: modifique o Mergesort.)

Questão 7 [CLSR, Ex 3.1-1, pg 50]

Sejam $f(n)$ e $g(n)$ funções assintoticamente não-negativas. Usando a definição básica da notação Θ , prove que

$$\max(f(n), g(n)) = \Theta(f(n) + g(n)).$$

Questão 8 [CLSR, Ex 3.1-2, pg 50]

Mostre que para quaisquer constantes reais a e b , tal que $b > 0$,

$$(n + a)^b = \Theta(n^b).$$

Questão 9 [CLSR, Ex 3.1-4, pg 50]

Mostre se:

- (a) $2^{n+1} \stackrel{?}{=} O(2^n)$.
- (b) $2^{2n} \stackrel{?}{=} O(2^n)$.

Questão 10 [CLSR, Ex 3.1-8, pg 50]

Pode-se estender a notação assintótica para o caso de dois parâmetros n e m que vão para infinito independentemente com taxas diferentes. Para uma dada função $g(n, m)$, denota-se $O(g(n, m))$ o conjunto de funções

$$O(g(n, m)) = \left\{ \begin{array}{l} f(n, m): \text{ existem constantes positivas } c, n_0, \text{ e } m_0 \text{ tais que} \\ 0 \leq f(n, m) \leq cg(n, m) \\ \forall n \geq n_0 \text{ e } m \geq m_0. \end{array} \right\}$$

Apresente as definições correspondentes para:

- (a) $\Omega(g(n, m))$
- (b) $\Theta(g(n, m))$

Questão 11 [CLSR, Ex 3.2-4, pg 57]

Mostre se:

- (a) A função $\lceil \log n \rceil!$ é limitada polinomialmente.
 (b) A função $\lceil \log \log n \rceil!$ é limitada polinomialmente.

Questão 12 [CLSR, Problema 3-2, pg 58]

(Crescimento assintótico relativo.) Indique, para cada par de expressões (A, B) na tabela abaixo, se A é O , o , Ω , ω , ou Θ de B . Assuma que $k \geq 1$, $\epsilon > 0$, e $c > 1$ são constantes. Sua resposta deve ser na forma SIM ou NÃO acompanhada da justificativa.

	A	B	O	o	Ω	ω	Θ
(a)	$\log^k n$	n^ϵ					
(b)	n^k	c^n					
(c)	$\sqrt{n}$	$n^{\sin n}$					
(d)	2^n	$2^{n/2}$					
(e)	$n^{\log m}$	$m^{\log n}$					
(f)	$\log(n!)$	$\log(n^n)$					

Questão 13 [CLSR, Problema 3-3, pg 58]

(Ordenação por taxa de crescimento assintótico.)

- (a) Classifique as funções abaixo pela ordem de crescimento, ou seja, ache uma permutação das funções $g_1, g_2, \dots, g_{30}$ que satisfaça a relação $g_1 = \Omega(g_2), g_2 = \Omega(g_3), \dots, g_{29} = \Omega(g_{30})$. Particione a lista em classes de equivalência tais que $f(n)$ e $g(n)$ estão na mesma classe se, e somente se, $f(n) = \Theta(g(n))$.

$\log(\log^* n)$	$2^{\log^* n}$	$(\sqrt{2})^{\log n}$	n^2	$n!$	$(\log n)!$
$(\frac{3}{2})^n$	n^3	$\log^2 n$	$\log(n!)$	2^{2^n}	$n^{\frac{1}{\log n}}$
$\ln \ln n$	$\log^* n$	$n \cdot 2^n$	$n^{\log \log n}$	$\log n$	1
$2^{\log n}$	$(\log n)^{\log n}$	e^n	$4^{\log n}$	$(n+1)!$	$\sqrt{\log n}$
$\log^*(\log n)$	$2^{\sqrt{2 \log n}}$	n	2^n	$n \log n$	$2^{2^{n+1}}$

A notação $\log^* n$ representa a função logarítmica “iterada” como definida na página 55 de CLRS.

- (b) Apresente um exemplo de uma função $f(n)$ não-negativa tal que para todas as funções $g_i(n)$ da letra anterior, $f(n)$ não é $O(g_i(n))$ nem $\Omega(g_i(n))$.

Questão 14 [CLSR, Ex 4.2-5, pg 72]

Use uma árvore de recursão para apresentar uma solução assintoticamente firme para a recorrência

$$T(n) = T(\alpha n) + T((1 - \alpha)n) + cn,$$

onde α é uma constante na faixa $0 < \alpha < 1$ e $c > 0$ é também uma constante.

Questão 15 [CLSR, Ex 4.3-1, pg 75]

Use o Teorema Mestre, se for possível, para apresentar limites assintóticos firmes para as seguintes recorrências:

- (a) $T(n) = 4T(\frac{n}{2}) + n$
- (b) $T(n) = 4T(\frac{n}{2}) + n^2$
- (c) $T(n) = 4T(\frac{n}{2}) + n^3$

Questão 16 [CLSR, Ex 4.3-2, pg 75]

A recorrência $T(n) = 7T(\frac{n}{2}) + n^2$ descreve o tempo de execução de um algoritmo A . Um algoritmo alternativo A' tem um tempo de execução $T'(n) = aT'(\frac{n}{4}) + n^2$. Qual é o maior número inteiro a que faz com A' seja assintoticamente mais rápido que A ?

Questão 17 [CLSR, Ex 4.3-4, pg 75]

O Teorema Mestre pode ser aplicado à recorrência $T(n) = 4T(\frac{n}{2}) + n^2 \log n$? Justifique a sua resposta. Apresente um limite superior assintótico para essa recorrência.

Questão 18 [CLSR, Problema 4-1, pg 85]

(*Exemplos de recorrência.*) Apresente os limites assintóticos superior e inferior para $T(n)$ em cada uma das recorrências abaixo. Assuma que $T(n)$ é constante para $n \leq 2$. Tente achar cada limite tão firme quanto possível e justifique a sua resposta.

- (a) $T(n) = 2T(\frac{n}{2}) + n^3$
- (b) $T(n) = T(\frac{9n}{10}) + n$
- (c) $T(n) = 16T(\frac{n}{4}) + n^2$
- (d) $T(n) = 7T(\frac{n}{3}) + n^2$
- (e) $T(n) = 7T(\frac{n}{2}) + n^2$
- (f) $T(n) = 2T(\frac{n}{4}) + \sqrt{n}$
- (g) $T(n) = T(n-1) + n$
- (h) $T(n) = T(\sqrt{n}) + 1$

Para cada um dos problemas abaixo, escreva o algoritmo, faça a implementação, testes e apresente o custo de complexidade identificando a operação considerada relevante. Lembre-se que a linguagem de programação é C. No caso de apresentar uma solução recursiva, discuta também e apresente a complexidade para o crescimento da pilha.

Questão 19 Fatorial de Números Grandes

Faça um programa que permita calcular o fatorial de números relativamente grandes como o fatorial de 10000. Você não deve usar qualquer biblioteca de suporte da linguagem C. O problema deve ser resolvido usando apenas a memória principal com a menor quantidade possível de espaço. O objetivo deste trabalho é estudar a complexidade de espaço. Procure implementar também a operação de multiplicação da forma mais eficiente possível. Procure na literatura algoritmos eficientes para multiplicação de números inteiros.

Sugestão: use um *nibble* (metade de um octeto ou byte) para armazenar um algarismo decimal ou, melhor ainda, algarismo hexadecimal.

Questão 20 Decomposição de Números

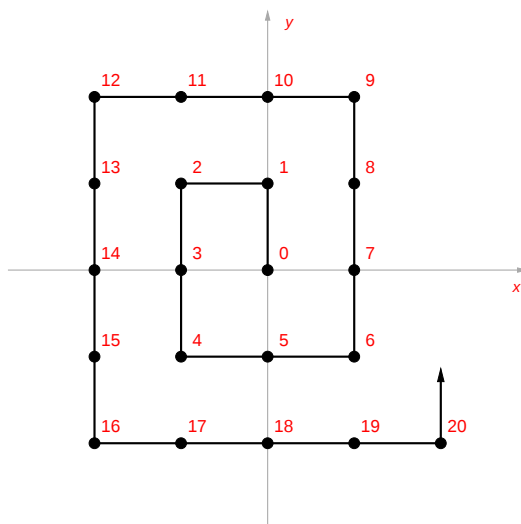
Faça um programa recursivo para gerar a decomposição de um número inteiro positivo na soma de todos os possíveis fatores como mostrado a seguir. Por exemplo, para $n = 5$, temos:

5
4 + 1
3 + 2
3 + 1 + 1
2 + 2 + 1
2 + 1 + 1 + 1
1 + 1 + 1 + 1 + 1

Questão 21 Espiral Quadrada

Seja a espiral quadrada como apresentada abaixo. Faça um programa que apresente as coordenadas (x, y) de um dado ponto n fornecido na entrada. Apresente três algoritmos distintos que executam no pior caso em:

- (a) $O(1)$
- (b) $O(\sqrt{n})$
- (c) $O(n)$
- (d) Você conhece algum problema “não usual” que tenha algoritmos com complexidades tão diferentes com o da Espiral Quadrada? Se sim, enuncie esse problema e indique os algoritmos e/ou referências para sua solução.



Questão 22 Máxima Soma

Dado um vetor com n números inteiros, determine a máxima soma encontrada em um sub-vetor contíguo desse vetor. Se todos números forem negativos assumir que a soma vale 0. A figura abaixo à esquerda mostra um vetor com 10 elementos. Nesse caso, a máxima soma é 187, dada pela soma dos elementos contíguos do sub-vetor de índices de 3 a 7, como mostrado na figura à direita.

Tente apresentar um algoritmo com custo de execução menor que $O(n^2)$.

31	-41	59	26	-53	58	97	-93	-23	84
----	-----	----	----	-----	----	----	-----	-----	----

31	-41	59	26	-53	58	97	-93	-23	84
		↑				↑			

Questão 23 Quadrado Mágico

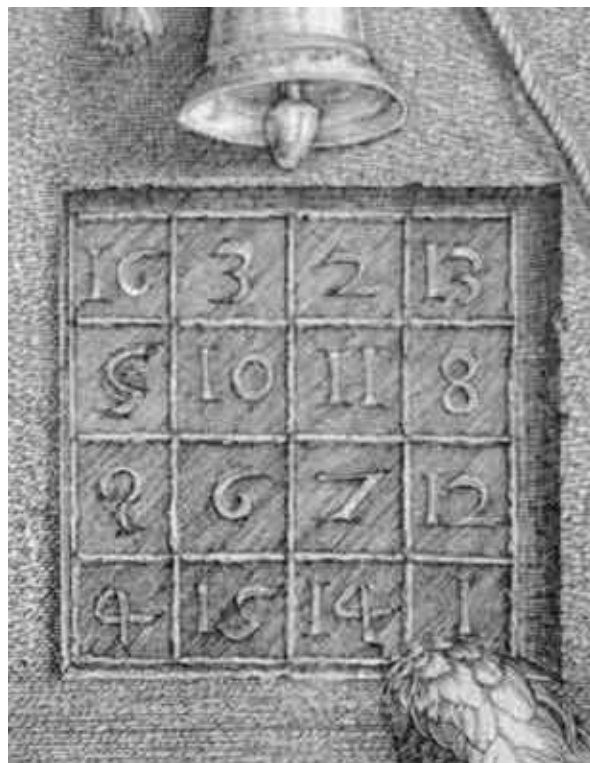
Quadrado Mágico é um quadrado de lado n , onde a soma dos números das linhas, das colunas e das diagonais é constante. Em cada posição do quadrado pode-se colocar um número entre 1 e n^2 , sendo que cada número só pode aparecer uma única vez.

A figura abaixo mostra uma possível solução para o quadrado mágico de lado $n = 3$.

2	7	6	→ 15
9	5	1	→ 15
4	3	8	→ 15
15	15	15	15

Gere o quadrado mágico para $3 \leq n \leq 10$.

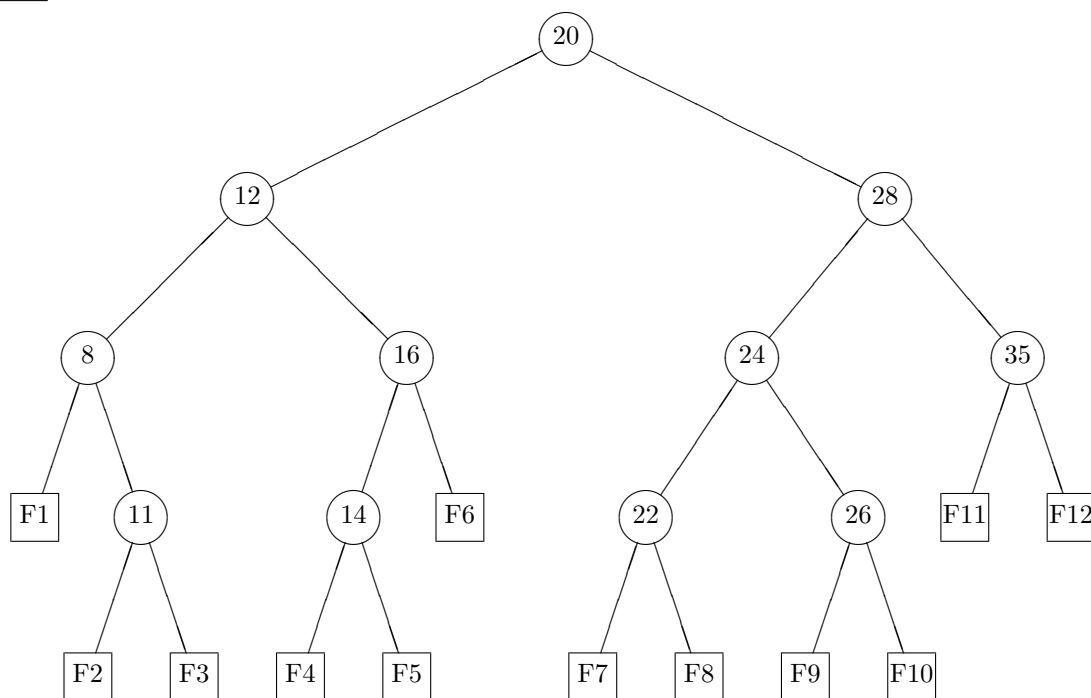
Quadrado Mágico e Arte. Um quadrado mágico de lado $n = 4$ aparece na obra “Melancolia I” (1514) de Albrecht Dürer, um dos maiores artistas alemães do Renascimento. Veja essa obra com o detalhe do quadrado mágico no lado direito.



“Melancolia I, como Dürer denomina a gravura que mostra um céu noturno iluminado apenas por um cometa, é o trabalho de maior conteúdo simbólico dentre todos os trabalhos deste artista. A mulher robusta refletindo sobre a fórmula que faz o mundo óbvio é não somente uma versão feminina de Fausto, comparável ao Jeremiah pintado por Michelangelo dois anos antes no teto da Capela Sistina, como também o oposto de Jerome em seu estudo. O santo dominou o caos que envolve Melancolia e colhe os frutos de seus esforços incansáveis, alcançando o ápice da vida. Melancolia ainda procura, em partes, pelo todo; ela ainda não atingiu a despreocupação de um estudo. Abandonado à inclemência da natureza durante a noite, o arco-íris anuncia sucesso. Acima da mulher, na parede sólida da casa, há uma balança, uma ampulheta, um sino e um quadro mágico, em que a soma dos números em qualquer direção é 34. A base do trabalho de Dürer era a adição e a mensuração exatas e, em Melancolia, são uma base segura para organizar uma nova estrutura para o mundo, a partir de utensílios de arquitetura e carpintaria. O bloco multifacetado, colossal, contrasta com a harmonia perfeita da esfera iluminada no primeiro plano. À

época, muito se teorizava sobre os quatro humores humanos: sanguíneo, fleumático, colérico e melancólico. Esta gravura em metal é um trabalho de arte de significância filosófica atemporal.” (Horst Michael, Albrecht Dürer – The Complete Engravings, Artline Editions, 1987.)

Questão 24 Caminhamentos em Árvore Binária de Pesquisa



Exemplo de uma árvore binária de pesquisa.

- (a) Escreva um procedimento recursivo baseado em um dos caminhamentos em árvore binária para calcular a altura dessa árvore.
- (b) Em cada entrada da tabela abaixo, diga SIM se a combinação linha/coluna é sempre verdadeira e NÃO, caso contrário.

	$\text{pré-ordem}(n) \prec \text{pré-ordem}(m)$	$\text{in-ordem}(n) \prec \text{in-ordem}(m)$	$\text{pós-ordem}(n) \prec \text{pós-ordem}(m)$
n está à esquerda de m			
n está à direita de m			
n é um ancestral de m			
n é um descendente de m			

A expressão $x \prec y$ significa que x precede y no caminhamento em questão. Um nó está à esquerda ou à direita de outro se e somente se eles têm um ancestral em comum.

- (c) O caminhamento por nível de uma árvore primeiro lista a raiz, depois todos os nós que estão no nível 1, depois todos os nós no nível 2, etc. Escreva um programa $O(n)$ (onde n é o número de nós na árvore) para listar todos os nós de uma árvore por nível (em cada nível, do nó mais à esquerda para o mais à direita).
- (d) Escreva um procedimento para listar todos os caminhos da raiz até os nós folhas.

Questão 25 Desafio: Conjectura de Goldbach

Em 1742, o matemático Prussiano Christian Goldbach escreveu uma carta para Leonhard Euler no qual ele propôs a seguinte conjectura: *Cada número par maior que 5 pode ser escrito como a soma de três primos.* Euler ficou

interessado no problema e respondeu com uma versão mais forte da conjectura: *Cada número par maior que 2 pode ser escrito como a soma de dois primos.*

A primeira conjectura é conhecida atualmente como conjectura “fraca” de Goldbach e a segunda como conjectura “forte” de Goldbach. Em geral, sem uma qualificação faz-se referência a versão forte. As duas questões continuam em aberto desde então.

A conjectura de Goldbach tem sido pesquisada por muitos matemáticos da área de teoria dos números. A maioria dos matemáticos acredita que a conjectura forte seja verdadeira, principalmente baseada em considerações estatísticas que tratam da distribuição de números primos: quanto maior o número par, mais provável é que possa ser escrito como a soma de dois primos.

Até 14/7/2008, um grupo de pesquisa liderado por Tomás Oliveira e Silva (<http://www.ieeta.pt/~tos/goldbach.html>), da Universidade de Aveiro em Portugal, já havia verificado a conjectura até 12×10^{17} , usando um programa de computador. Observe que aparentemente isto pode parecer uma “bobagem” mas existem vários problemas computacionais interessantes que devem ser resolvidos para poder testar a conjectura para números desta ordem de grandeza.

O Desafio. Achar o maior p tal que $S(p) \geq 10^{18}$ (partições de Goldbach). Para mais detalhes veja a referência acima. Como isso pode ser feito de forma isolada e cooperativa?

Questão 26 Trabalho Prático: Operações Aritméticas com “Números Grandes”

Objetivo. Projetar algoritmos eficientes, em termos de tempo e espaço, para executar as quatro operações aritméticas básicas (+, ×, −, /) de números inteiros que possuem entre 50 000 e 100 000 algarismos.

Entrada dos dados. Cada operação a ser executada deve ser fornecida em um arquivo texto contendo três linhas: a primeira deve conter o operador aritmético e as duas seguintes os operandos, ou seja, a entrada de dados segue a notação pré-fixada. Assim, a operação aritmética

$$A \times B$$

deve ser fornecida como

1ª linha:	×
2ª linha:	A
3ª linha:	B

Linguagem de programação. Este trabalho deve ser feito obrigatoriamente na linguagem C padrão.

Documentação. Os seguintes documentos devem ser preparados:

1. Uma breve descrição sobre o projeto da solução proposta (duas a três páginas);
2. Discussão da complexidade de tempo e espaço da solução proposta.

Observação. Outras informações sobre o processo de submissão do trabalho, lista completa do que deve ser entregue, data de entrega, etc, serão disponibilizadas na página da disciplina.