Algoritmos e Estruturas de Dados II

Exercícios

7 de dezembro de 2024

1. Escreva um algoritmo *recursivo* para resolver o problema de calcular o fatorial[1] de um inteiro n dado, isto é, uma solução para o seguinte problema computacional.

   Fatorial (FAT)

   Instância: $n \in \mathbb{N}$.
   Resposta: $n!$

2. Proponha um algoritmo *recursivo* para contar o número de ocorrências de um valor dado em um vetor, isto é, uma solução para o seguinte problema computacional.

   Contagem de Ocorrências em Vetor (COV)

   Instância:
   (v, a, b, x), onde v é um vetor indexado por $[a..b]$ e x é um valor.
   Resposta:
   O número de ocorrências de x em $v[a..b]$, isto é, o valor de

   $$|\{i \in [a..b] \mid v[i] = x\}| .$$

3. O algoritmo Troca(v, a, b) abaixo troca entre si os conteúdos das posições de $v[a]$ e $v[b]$.

   Troca(v, a, b)

   $x \leftarrow v[a]$
   $v[a] \leftarrow v[b]$
   $v[b] \leftarrow x$

   Considere problema computacional de reverter um vetor dado.

   Reversão (REV)

   Instância: (v, a, b), onde v é um vetor indexado por $[a..b]$.
   Resposta: O vetor v revertido, isto é, modificado de tal forma que o primeiro elemento se torna o último, o segundo se torna o penúltimo e assim por diante.

   Escreva um algoritmo *recursivo* para resolver este problema, usando o algoritmo Troca como subrotina.

[1] Por exigência da ANVISA este algoritmo precisa aparecer como exemplo ou exercício da disciplina ...

4. Dados $n \in \mathbb{N}$ e $k \leq n$, o *fatorial descendente* n_k é definido como

$$n_k := n(n-1)(n-2)\dots(n-k+1) = \prod_{i=n-k+1}^{n} i$$

Considere o problema de calcular o fatorial descendente[2], isto é,

Fatorial Descendente (FatD)
Instância: (n, k), onde $n \in \mathbb{N}$ e $k \geq 0$.   Resposta: n_k

Escreva um algoritmo *recursivo* para resolver o problema.

5. Dizemos que o vetor $v[a..b]$ é um *palíndromo* se a leitura de v na ordem direta é igual à sua leitura na ordem reversa, isto é, se

$$(v[a], v[a+1], \dots, v[b-1], v[b]) = (v[b], v[b-1], \dots, v[a+1], v[a]).$$

Escreva um algoritmo *recursivo* para resolver o problema de decidir se um vetor é um palíndromo, isto é, uma solução para o seguinte problema computacional.

Palíndromo (Pal)
Instância: (v, a, b), onde v é um vetor indexado por $[a..b]$.   Resposta: sim se $v[a..b]$ é um palíndromo ou não, caso contrário

6. Seja p um vetor de números racionais indexado por $[a..b]$. Dados $c,\ d \in [a..b]$ com $c \leq d$, vamos denotar por $p_{c,d}(x)$ o polinômio

$$p_{c,d}(x) := p[c]x^0 + p[c+1]x^1 + \ldots + p[d]x^{d-c} = \sum_{i=c}^{d} p[i]x^{i-c}$$

Por exemplo, se p é o vetor dado por

i	0	1	2	3	4	5
$p[i]$	4	8	15	16	23	42

então,

$$p_{0,5}(x) = 4x^0 + 8x^1 + 15x^2 + 16x^3 + 23x^4 + 42x^5,$$

[2]Observe que como $n! = n_n$, este problema é uma generalização do problema Fatorial do Exercício 1.

e

$$p_{2,4}(x) = 15x^0 + 16x^1 + 23x^2.$$

Proponha um algoritmo *recursivo* para o problema de, dados um polinômio p e um valor $x \in \mathbb{Q}$, avaliar $p(x)$, isto é, ua solução para o seguinte problema computacional.

Avaliação de Polinômio (Pol)	
Instância:	(p, a, b, x), onde p é um vetor de números racionais indexado por $[a..b]$ e x é um número racional.
Resposta:	$p_{a,b}(x)$.

Use o Algoritmo $\mathsf{Exp}(x, n)$ discutido em aula como subrotina.

7. A *sequência de Fibonacci* é a função $F\colon \mathbb{N} \to \mathbb{N}$ dada pela recorrência

$$F(n) = \begin{cases} n, & \text{se } n \leq 1, \\ F(n-1) + F(n-2), & \text{se } n > 1. \end{cases}$$

   (a) Escreva um algoritmo *recursivo* para computar o n-ésimo número da Sequência de Fibonacci, isto é, uma solução para o seguinte problema computacional.

Número de Fibonacci (Fib)	
Instância:	$n \in \mathbb{N}$.
Resposta:	$F(n)$.

   (b) Seja $S(n)$ o número de somas efetuadas pela execução de seu algoritmo sobre a instância n por meio de uma recorrência.

8. O Problema das Torres de Hanói consiste de três torres, A, B e C onde é possível empilhar discos.

   Inicialmente uma das torres tem discos de tamanhos (dois a dois) distintos empilhados de tal forma que o tamanho de cada disco é *menor* do que o tamanho do disco sobre o qual ele repousa. As demais estão vazias.

   O objetivo é transferir todos os discos desta torre para outra (usando a terceira quando necessário) por meio de uma sequência de *movimentos* obedecendo as seguintes regras.

   (a) Cada *movimento* consiste em mover um disco do topo de uma torre para o topo de outra, e

(b) é proibido colocar um disco sobre outro de tamanho menor.

Proponha um algoritmo *recursivo* Hanoi(n, A, B, C), que recebe um inteiro n e escreve uma solução para a instância do Problema das Torres de Hanói onde inicialmente há n discos empilhados na Torre A que devem ser transferidos para a Torre B.

A solução deve ser um algoritmo formado por uma sequência de instruções da forma "$x \to y$" cada uma delas significando, "mova o disco no topo da Torre x para o topo da Torre y".

9. O *Método de Horner* para a solução do problema de Avaliação de Polinômio (descrito no Exercício 6) é um algoritmo baseado na observação de que, dada uma instância (p, a, b, x) do problema, então

$$p_{a,b}(x) = \begin{cases} 0, & \text{se } a > b, \\ p[a] + xp_{a+1,b}(x), & \text{se } a \leq b. \end{cases}$$

   Escreva um algoritmo *recursivo* para resolver o problema de Avaliação de Polinômio usando o Método de Horner.

10. Seja $m(n)$ o número de multiplicações efetuadas pelo algoritmo do Exercício 1 para computar a instância n.

    (a) Expresse $m(n)$ como uma recorrência.

    (b) Resolva esta recorrência.

11. Seja $c(n)$ o número de comparações com elementos de v efetuadas pelo algoritmo do Exercício 2 para computar a instância $(v, a, a+n-1, x)$.

    (a) Expresse $c(n)$ como uma recorrência.

    (b) Resolva esta recorrência.

12. Seja $t(n)$ o número de execuções do Algoritmo Troca na execução do algoritmo do Exercício 3 para computar a instância $(v, a, a+n-1)$.

    (a) Expresse $t(n)$ como uma recorrência.

    (b) Resolva esta recorrência.

13. Seja $m(n, k)$ o número de multiplicações efetuadas pelo algoritmo do Exercício 4 para computar a instância (n, k).

(a) Expresse $m(n,k)$ como uma recorrência.

(b) Expresse $m(n,n)$ como uma recorrência.

(c) Resolva esta recorrência.

(d) Prove[3] que $m(n,k) = n - k$ para todo $0 \leq k \leq n$.

(e) Use a resposta do item anterior para obter uma expressão não recorrente para $m(n,k)$.

14. Seja $c(v,a,b)$ o número de comparações entre elementos de v efetuadas na execução do algoritmo do Exercício 5 para a instância (v,a,b) do problema, e sejam

$$
\begin{aligned}
c^+(n) &:= \max\{c(v,a,a+n-1) \mid (v,a,a+n-1) \text{ é instância de } \mathsf{Pal}\},\\
c^-(n) &:= \min\{c(v,a,a+n-1) \mid (v,a,a+n-1) \text{ é instância de } \mathsf{Pal}\}.
\end{aligned}
$$

(a) Descreva as instâncias $(v,a,a+n-1)$ para as quais

$$c(v,a,b) = c^-(n).$$

(b) Descreva as instâncias $(v,a,a+n-1)$ para as quais

$$c(v,a,b) = c^+(n).$$

(c) Dê uma expressão para $c^-(n)$.

(d) Expresse $c^+(n)$ como uma recorrência.

(e) Resolva esta recorrência.

15. Seja $m(n)$ o número de multiplicações efetuadas pelo algoritmo do Exercício 6 para computar a instância $(p,a,a+n-1,x)$.

(a) Expresse $m(n)$ por uma recorrência.

(b) Resolva esta recorrência.

16. Seja $m(n)$ o número de vezes que o algoritmo do Exercício 8 escreve uma instrução do tipo "$x \to y$".

(a) Expresse $m(n)$ por meio de uma recorrência.

(b) Prove que $m(n) = 2^n - 1$, para todo $n \geq 0$.

[3] **Sugestão:** indução em $n - k$.

17. Seja $h(n)$ o número de multiplicações efetuadas pelo algoritmo do Exercício 9 para computar a instância $(p, a, a+n-1, x)$.

    (a) Expresse $h(n)$ por meio de uma recorrência.

    (b) Resolva esta recorrência.

    (c) Compare com o valor de $m(n)$ no Exercícios 15 e prove que

    $$\lim \frac{h(n)}{m(n)} = 0.$$

18. Considere o problema de Avaliação de Polinômio descrito no Exercício 6.

    (a) Proponha um algoritmo *recursivo* para o problema baseado na observação de que se $a \leq b$, então

    $$p_{a,b}(x) = p_{a,m}(x) + x^{m+1-a} p_{m+1,b}(x),$$

    onde

    $$m = \frac{a+b}{2}.$$

    (b) Seja $m(n)$ o número de multiplicações feitas pelo seu algoritmo para computar a instância $(v, a, a+n-1, x)$. Expresse $m(n)$ por meio de uma recorrência.

    (c) Resolva esta recorrência.

19. Dados $n, k \in \mathbb{N}$ o *coeficiente binomial* $\binom{n}{k}$ é o número de subconjuntos de k elementos de um conjunto com n elementos. Observe que a partir desta definição, $\binom{n}{k} = 0$ sempre que $k \notin [0..n]$.

    O problema de computar o coeficiente binomial pode ser formulado como segue.

Coeficiente Binomial (CB)
Instância: (n, k), onde $n, k \in \mathbb{N}$. Resposta: $\binom{n}{k}$

    Este exercício pede que você proponha diferentes algoritmos para resolver este problema, analise-os e compare-os entre si.

    (a) Proponha um algoritmo recursivo para o problema do Coeficiente Binomial que usa o algoritmo do Exercício 1 como subrotina e o fato de que

    $$\binom{n}{k} = \frac{n!}{k!(n-k)!}, \text{ para todo } 0 \leq k \leq n.$$

(b) Use a resposta do Exercício 10 para obter uma expressão para o número de operações aritméticas efetuadas pelo algoritmo do item anterior para computar a instância (n, k) do problema.

(c) Proponha um algoritmo recursivo para o problema do Coeficiente Binomial que usa o algoritmo do Exercício 4 como subrotina e o fato de que

$$\binom{n}{k} = \frac{n_k}{k!}, \text{ para todo } 0 \leq k \leq n.$$

(d) Use a resposta do Exercício 13 para obter uma expressão para o número de operações aritméticas efetuadas pelo algoritmo do item anterior para computar a instância (n, k) do problema.

(e) Dados $n \in \mathbb{N}$ e $0 \leq k \leq n$ seja

$$Q(n, k) := \frac{\binom{n}{k}}{\binom{n-1}{k-1}}.$$

Observe que a expressão acima fornece outra recorrência para o cálculo de $\binom{n}{k}$, a saber,

$$\binom{n}{k} = Q(n, k)\binom{n-1}{k-1}, \text{ para todo } 0 \leq k \leq n.$$

Proponha um algoritmo *recursivo* para o problema do Coeficiente Binomial que usa esta igualdade.

(f) Para cada $n \in \mathbb{N}$ seja $A(n, k)$ o número de operações aritméticas efetuadas pelo algoritmo do item anterior para computar a instância (n, k) do problema. Expresse $A(n, k)$ por meio de uma recorrência.

(g) Explique o que muda nas respostas dos itens anteriores se os algoritmos incorporarem a seguinte propriedade dos coeficientes binomiais.

$$\binom{n}{k} = \binom{n}{n-k}, \text{ para todo } n, k \in \mathbb{N}.$$

20. Sejam $a \leq b \in \mathbb{Z}$ e seja $m = \left\lfloor \frac{a+b}{2} \right\rfloor$. Prove que

(a) $|[a..m]| = \left\lceil \frac{n}{2} \right\rceil$

(b) $|[m+1..b]| = \left\lfloor \frac{n}{2} \right\rfloor$

21. Considere o seguinte algoritmo.

```
Soma_2(v, a, b)
---
    Se a > b
        Devolva 0
    m ← ⌊(a+b)/2⌋
    Devolva Soma_2(v, a, m) + Soma_2(v, m + 1, b)
```

(a) Seja $s(n)$ o número de somas efetuadas na execução de $\mathsf{Soma}_2(v, a, a+n-1)$. Expresse $s(n)$ por meio de uma recorrência.

(b) Resolva esta recorrência.

22. Considere o seguinte algoritmo para o problema de Busca em Vetor.

```
B(x,v,a,b)
---
    Se a > b
        Devolva não
    m ← ⌊(a+b)/2⌋
    Se x = v[m]
        Devolva m
    r ← B(x, v, a, m − 1)
    Se r ≠ não
        Devolva r
    Devolva B(x, v, m + 1, b)
```

(a) Execute $\mathsf{B}(x, v, a, b)$ para as mesmas instâncias do problema de Busca em Vetor usadas como exemplo em aula.

(b) Seja $c(x, v, a, b)$ o número de comparações entre elementos de v efetuadas na execução de $\mathsf{B}(x, v, a, b)$, e seja

$$c^+(n) = \max\{c(x, v, a, b) \mid b - a + 1 = n\}.$$

i. Descreva um conjunto de instâncias (x, v, a, b) do problema para as quais temos

$$c(x, v, a, b) = c^+(n)?$$

ii. Expresse $c^+(n)$ como uma recorrência[4].

iii. Calcule o valor de $c^+(n)$ para $n \in [0..16]$.

[4] **Sugestão:** use o Exercício 20.

iv. Com base nos valores obtidos no item 22(b)iii, formule uma hipótese para a solução da recorrência do item 22(b)ii.

v. Prove por indução que sua solução do item 22(b)iv está correta.

23. Considere o seguinte problema computacional.

Ponto Fixo de Vetor Ordenado (PFVO)
Instância: (v, a, b), onde $v[a..b]$ é um vetor ordenado.
Resposta: $m \in [a..b]$ tal que $m = v[m]$ ou $a - 1$ caso não exista tal m.

(a) Escreva um algoritmo recursivo para este problema.

(b) Quantas comparações com elementos de v faz o seu algoritmo no melhor e no pior caso, em função do número $n = b - a + 1$ de elementos do vetor?

(c) Escreva uma versão iterativa de seu algoritmo.

24. Como a precisão de qualquer dispositivo computacional é finita, o cálculo computacional do valor de uma função com contra-domínio em $\mathbb{R}$ é quase sempre uma aproximação.

Dado $\varepsilon > 0$, dizemos que um algoritmo F calcula uma ε-aproximação da função $f\colon D \to \mathbb{R}$ se F é um algoritmo que recebe um número $x \in D$ como (parte da) entrada e devolve $y \in \mathbb{Q}$ satisfazendo $|f(x) - y| \leq \varepsilon$.

Considere o seguinte problema.

Raiz Quadrada (RQ)
Instância: (x, ε, a, b), onde x, a e b são números não-negativos satisfazendo $\sqrt{x} \in [a, b]$ e $\varepsilon > 0$.
Resposta: Uma ε-aproximação de $\sqrt{x}$, isto é, um número $y \in [a, b] \subseteq \mathbb{Q}$ satisfazendo $|\sqrt{x} - y| \leq \varepsilon$.

(a) Baseado na idéia de busca binária, escreva um algoritmo recursivo que só usa operações aritméticas elementares[5] para resolver este problema.

(b) Seja $A(l, \varepsilon)$ o número de operações aritméticas elementares feitas pela execução de seu algoritmo para a instância $(x, \varepsilon, a, a + l)$. Expresse $A(l, \varepsilon)$ por meio de uma recorrência.

(c) Resolva esta recorrência.

[5] Isto é, somas, subtrações, multiplicações e divisões.

(d) Conclua que a execução de seu algoritmo com a instância $A(x, \varepsilon, a, b)$ faz no máximo $4\left(\lceil \lg((b-a)/\varepsilon) \rceil + 1\right)$ operações aritméticas elementares.

(e) Escreva um Algoritmo recursivo que recebe como entrada $x \geq 0$ e $\varepsilon > 0$ e devolve uma ε-aproximação de $\sqrt{x}$ fazendo no máximo $4\left(\lceil \lg |x-1| / \varepsilon \rceil + 1\right)$ operações aritméticas elementares.

25. Considere o seguinte problema computacional.

Raiz Cúbica (RC)	
Instância:	(x, ε, a, b), onde x, a e b são números racionais não-negativos satisfazendo $\sqrt[3]{x} \in [a, b]$ e ε é um número racional positivo.
Resposta:	Uma ε-aproximação[a] de $\sqrt[3]{x}$, isto é, um número $r \in [a, b] \subseteq \mathbb{Q}$ satisfazendo $\lvert \sqrt[3]{x} - r \rvert \leq \varepsilon$.

[a] Veja o Exercício 24.

(a) Baseado na idéia de busca binária, escreva um algoritmo recursivo que só usa operações aritméticas elementares para resolver este problema.

(b) Formule uma versão iterativa deste algoritmo.

(c) Seja $P(l, \varepsilon)$ a profundidade (máxima) de recursão na execução de seu algoritmo para a instância $(x, \varepsilon, a, a+l)$.

   i. Descreva $P(l, \varepsilon)$ por meio de uma recorrência.

   ii. Prove que $P(l, \varepsilon) = \lceil \lg l/\varepsilon \rceil + 1$.

(d) Seja $A(l, \varepsilon)$ o número de operações aritméticas elementares envolvendo números racionais feitas pela execução de seu algoritmo para a instância $(x, \varepsilon, a, a+l)$. Expresse $A(l, \varepsilon)$ por meio de uma recorrência.

(e) Estabeleça um limitante superior para $A(l, \varepsilon)$.

26. Considere o seguinte problema.

Logaritmo (Log)	
Instância:	(x, ε, a, b), onde x, a e b são números racionais não-negativos satisfazendo $\log x \in [a, b]$ e ε é um número racional positivo.
Resposta:	Uma ε-aproximação[a] de $\log x$, isto é, um número $l \in [a, b]$ satisfazendo $|\log x - l| \leq \varepsilon$.

[a] Veja o Exercício 24.

(a) Baseado na idéia de busca binária, escreva um algoritmo recursivo que só usa operações aritméticas elementares e exponenciação para resolver este problema.

(b) Formule uma versão iterativa deste algoritmo.

(c) Seja $P(l, \varepsilon)$ a profundidade (máxima) de recursão na execução de seu algoritmo para a instância $(x, \varepsilon, a, a + l)$.

   i. Descreva $P(l, \varepsilon)$ por meio de uma recorrência.

   ii. Prove que $P(l, \varepsilon) = \lceil \lg l/\varepsilon \rceil + 1$.

(d) Seja $A(l, \varepsilon)$ o número de operações aritméticas (inclusive exponenciação) envolvendo números racionais feitas pela execução de seu algoritmo para a instância $(x, \varepsilon, a, a + l)$. Expresse $A(l, \varepsilon)$ por meio de uma recorrência.

(e) Estabeleça um limitante superior para $A(l, \varepsilon)$.

27. Dizemos que p é um *ponto fixo* da função $f \colon D \to \mathbb{R}$ se $f(p) = p$.

Do estudo do Cálculo sabemos que se $f \colon D \to \mathbb{R}$ é contínua em $[a, b]$ com $f(a) < a$ e $f(b) > b$, então f tem ponto fixo em $[a, b]$.

Considere o seguinte problema computacional.

Ponto Fixo de Função Contínua (PFFC)

Instância: (F, ε, a, b), onde F é um algoritmo que calcula uma ε-aproximação (veja o Exercício 24) da função contínua $f \colon D \to \mathbb{R}$ satisfazendo

$$\begin{aligned} F(a) &< a, \\ F(b) &> b. \end{aligned}$$

Resposta:
Uma ε-aproximação de um ponto fixo de f em $[a, b]$.

(a) Escreva um algoritmo recursivo baseado na ideia de busca binária para resolver este problema.

(b) Dado $l > 0$, expresse o número de execuções de F efetuadas pelo seu algoritmo para computar a instância $(x, \varepsilon, a, a + l)$ por meio de uma recorrência.

(c) Resolva esta recorrência.

28. Uma *equação* é uma expressão da forma $f(x) = g(x)$ onde $f, g \colon \mathbb{R} \to \mathbb{R}$. Uma *solução da equação* $f(x) = g(x)$ é um número $s \in \mathbb{R}$ satisfazendo $f(s) = g(s)$.

Considere o seguinte problema computacional.

Solução de Equação (SE)

Instância: $(F, G, \varepsilon, a, b)$, onde F e G são algoritmos que calculam ε-aproximações (veja o Exercício 24) de funções contínuas $f, g \colon \mathbb{R} \to \mathbb{R}$ respectivamente, e $\varepsilon, ab \in \mathbb{Q}$ satisfazem

$$\begin{aligned} F(a) &\leq G(a), \\ F(b) &\geq G(b). \end{aligned}$$

Resposta: Uma ε-aproximação de uma solução da equação $f(x) = g(x)$.

Observe que

(a) o Exercício 24 pede um algoritmo que devolva uma (ε-aproximação) da equação $x^2 =$

algoritmo que recebe um valor x como entrada e devolve uma solução aproximada da equação $f(X) = g(X)$ onde $f(X) = X^2$ e $g(X) = x$ (isto é $g(X) = x$ para todo $X \in \mathbb{R}$). Do mesmo modo, o Exercício 25 pede um algoritmo que recebe x como entrada e devolve uma solução aproximada da equação $X^3 = x$, o Exercício 26 pede um algoritmo que recebe x como entrada e devolve uma solução aproximada da equação $10^X = x$ e o Exercício 27 pede um algoritmo que recebe F e x como entrada e devolve uma solução aproximada da equação $F(X) = X$.

O seguinte problema computacional generaliza os problemas dos Exercícios 24, 25, 26 e 27.

(a) Escreva um algoritmo recursivo para resolver este problema.

(b) Expresse o número de execuções de F e G efetuadas pelo seu algoritmo em função dos valores de $F(m)$, $F(M)$, $G(m)$, $G(M)$ e ε.

(c) Escreva uma versão iterativa de seu algoritmo. Qual o invariante da iteração?

29. Uma *raiz* ou *zero* de uma função $f\colon D \to \mathbb{R}$ é um valor $x \in D$ tal que $f(x) = 0$.

    Considere o seguinte problema computacional.

    Zero de Função Não Decrescente (ZFND)

    Instância: (F, ε, a, b), onde F é um algoritmo que calcula uma ε-aproximação de uma função contínua não decrtescente $f\colon D \to \mathbb{R}$ e $\varepsilon, a, b \in \mathbb{Q}$ satisfazem

    $$\begin{aligned} F(a) &< 0, \\ F(b) &> 0. \end{aligned}$$

    Resposta: Uma ε-aproximação de um zero da função f em $[a..b]$.

    (a) Escreva um algoritmo recursivo para resolver este problema baseado na ideia de busca binária.

    (b) Dado $l > 0$, expresse o número de execuções de F efetuadas pelo seu algoritmo para computar a instância $(x, \varepsilon, a, a + l)$ por meio de uma recorrência.

    (c) Resolva esta recorrência.

    (d) Explique como reduzir os problemas computacionais dos Exercícios 24, 25, 26, 27 e 28 ao problema de Zero de Função Contínua.

30. Considere o seguinte problema computacional.

    Zero de Função Contínua (ZFC)

    Instância: (F, ε, a, b), onde F é um algoritmo que calcula uma ε-aproximação[a] de uma função contínua $f\colon D \to \mathbb{R}$ e $a, b \in D$ satisfazendo $F(a)F(b) < 0$.

    Resposta: Uma ε-aproximação de um zero[b] da função f em $[a, b]$.

    [a] Veja o Exercício 24

    [b] Veja o Exercício 29

(a) Escreva um algoritmo recursivo para resolver este problema baseado na ideia de busca binária.

(b) Dado $l > 0$, expresse o número de execuções de F efetuadas pelo seu algoritmo para computar a instância $(x, \varepsilon, a, a + l)$ por meio de uma recorrência.

(c) Resolva esta recorrência.

(d) Explique como reduzir o problema computacional do Exercício 28 ao problema de Zero de Função Contínua.

31. Descreva as instâncias (v, a, b) para as quais o algoritmo Ordena_I faz

    (a) o menor número possível de trocas com elementos de v,

    (b) o maior número possível de trocas com elementos de v.

32. Escreva uma versão recursiva do Algoritmo Insere discutido em aula, isto é, um algoritmo recursivo para o seguinte problema.

Inserção em Vetor Ordenado (IVO)	
Instância:	(v, a, b) onde $v[a..b-1]$ é um vetor ordenado.
Resposta:	o vetor v modificado de tal forma que $v[a..b]$ é um vetor ordenado.

    (a) Seja $T^+(n)$ o número máximo de trocas efetuado pelo seu algoritmo para computar uma instância com $n = b - a + 1$. Descreva $T^+(n)$ através de uma recorrência.

    (b) Resolva esta recorrência.

33. Uma *cópia de memória* é o que acontece quando uma variável é copiada para outra durante a execução de um algoritmo. Por exemplo, o algoritmo Troca, abaixo, realiza 3 cópias de memória.

    Troca(v, a, b)

    $x \leftarrow v[a]$
    $v[a] \leftarrow v[b]$
    $v[b] \leftarrow x$

    O Algoritmo Insere discutido em aula usa o Algoritmo Troca acima. Na análise feita em aula vimos que o algoritmo Insere faz $n-1$ trocas entre elementos do vetor no pior caso para uma instância com $n = b-a+1$, o que corresponde a $3(n-1)$ cópias de memória.

(a) Quantas cópias de memória faz o algoritmo Ordena_I (discutido em aula) no pior caso para uma instância (v, a, b) com $n = b - a + 1$?

(b) Modifique o Algoritmo Insere de tal forma que ele faça no máximo $n + 1$ cópias de memória para uma instância (v, a, b) com $n = b - a + 1$.

(c) Quantas cópias de memória faz o algoritmo Ordena_I discutido em aula no pior caso, para uma instância (v, a, b) com $n = b - a + 1$, usando esta versão modificada do Algoritmo Insere?

34. A partir do fato de que

$$\sum_{i=1}^{n} \lg n \sim n \lg n,$$

prove [6] que

$$\sum_{i=1}^{n} \lfloor \lg n \rfloor \sim n \lg n \sim \sum_{i=1}^{n} \lceil \lg n \rceil .$$

35. Escreva o Algoritmo $\mathsf{Ordena}(v, n)$ que ordena o vetor $v[1..n]$ por intercalação ($\mathsf{MergeSort}$) mas *sem uso de recursão.* Você pode usar assumir que

(a) o valor de n é uma potência de 2.

(b) está "disponível" o Algoritmo $\mathsf{Intercala}(v, a, m, b)$ que efetua a intercalação dos vetores ordenados $v[a..m]$ e $v[m + 1..b]$ tal como discutido em aula.

(c) seu algoritmo não precisa fazer as intercalações de vetores na mesma ordem que o algoritmo recursivo.

36. Que modificações devem ser feitas no Algoritmo Ordena_Q (QuickSort) para que ordene em ordem não-crescente?

37. Na execução de $\mathsf{Ordena}_Q(v, a, a + n - 1)$ (QuickSort), qual o número máximo de vezes em que um mesmo elemento de valor máximo em v pode ser movido?

38.

[6] **Sugestão:** Use o fato de que

$$\lg n - 1 < \lfloor \lg n \rfloor \leq \lg n \leq \lceil \lg n \rceil < \lg n + 1, \text{ para todo } n \geq 1.$$

39. Seja $(v, 1, n)$ uma instância do problema de ordenação onde os elementos de $v[1..n]$ são os inteiros de 1 a n. Suponha que cada execução do *Algoritmo* Particiona(v, a, b) durante a execução do Algoritmo Ordena$_Q(v, 1, n)$ (QuickSort) sobre esta instância devolve o valor $a + 1$.

   (a) Apresente uma instância assim com $n = 10$.

   (b) Expresse o número de comparações na execução de Ordena$_Q(v, 1, n)$ sobre esta instância em função de n.

40. Apresente seis vetores de tamanho 10, com os valores pertencentes ao conjunto [1..10], onde o Algoritmo Ordena$_Q$ (Quick Sort) efetua o maior número possível de comparações.

41. Quantas comparações faz o Algoritmo Ordena$_Q(v, a, a+n-1)$ (QuickSort) faz ao ordenar um vetor v com todos os valores iguais?

42. Qual é a profundidade da recursão do Algoritmo Ordena$_Q$ (QuickSort) no melhor caso e pior caso?

43. Execute o Algoritmo MontaHeap$(v, 11)$ onde $v[1..11]$ é o vetor

i	1	2	3	4	5	6	7	8	9	10	11
$v[i]$	1	2	3	4	5	6	7	8	9	10	11

   mostrando o conteúdo do vetor v ao início do laço em cada iteração.

44. Elabore uma versão iterativa para o Algoritmo ConsertaHeap(v, i, k) discutido em aula.

45. Na versão de HeapSort discutida em aula, o *heap* foi implementado num vetor indexado por $[1..n]$.

    Modifique os algoritmos de forma que o vetor não precise ser indexado a partir de 1.

46. A partir da versão de ConsertaHeap(v, i, a, b) proposta na resposta do Exercício 45, elabore versões recursivas para MontaHeap(v, a, b) e Ordena$_H(v, a, b)$.

47. Elabore uma versão iterativa para o Algoritmo ConsertaHeap(v, i, a, b) dado como resposta para o Exercício 45.

48. Escreva um algoritmo para o seguinte problema computacional.

2-decomposição (2DEC)
Instância:
(x, v, a, b) onde x é um valor e v é um vetor indexado por $[a..b]$.
Resposta:
Um par de índices $\{p, q\} \subseteq [a..b]$ distintos satisfazendo $v[a] + v[b] = x$ ou $\emptyset$ caso não existam tais índices.

Seu algoritmo deve fazer assintoticamente um máximo de $n \lg n$ comparações e n somas para qualquer instância do problema

49. O seguinte algoritmo é uma formulação do "Método da Bolha" (BubbleSort) para ordenação.

```
OrdenaBolha(v, a, b)
---
Se a = b
    Termine
Bolha(v, a, b)
OrdenaBolha(v, a + 1, b)
```

```
Bolha(v, a, b)
---
Para i ← b até a + 1
    Se v[i] < v[i − 1]
        Troca(v, i, i − 1)
```

(a) Execute o Algoritmo OrdenaBolha$(v, 1, 6)$ onde $v[1..6]$ é o vetor

i	1	2	3	4	5	6
$v[i]$	16	23	4	42	15	8,

mostrando o conteúdo do vetor e os valores de a e b ao início de cada execução de OrdenaBolha(v, a, b).

(b) Dê uma expressão para $C_B(n)$, o número de comparações na execução de Bolha$(v, a, a + n - 1)$.

(c) Expresse $C(n)$, o número de comparações na execução de OrdenaBolha$(v, a, a + n - 1)$, por meio de uma recorrência.

(d) Resolva a recorrência do item anterior.

(e) Dê uma expressão para $T_B^-(n)$, o número mínimo de execuções de Troca() na execução de OrdenaBolha$(v, a, a + n - 1)$.

(f) Dê uma expressão para $T_B^+(n)$, o número máximo de execuções de Troca() na execução de OrdenaBolha$(v, a, a + n - 1)$.