IFMG Campus Formiga **Matemática Computacional**

Lista: Aquecimento (*Warm Up*)

Prof. Diego Mello Março/2019

Nome: _____ Nota: _____

INSTRUÇÕES

Leia atentamente a lista antes de resolvê-la. A lista é individual e deve ser entregue em formato digital até a data limite, combinada com o professor da disciplina nestas instruções.

Todos os exercícios devem conter, sem excessão, um cabeçalho indicando o nome do aluno, matrícula, curso e exercicio correspondente. O arquivo deverá ser salvo com o nome indicado na frente do exercício, entre parênteses (por exemplo, `sqrt.c` para o programa que utiliza a equação de recorrência de Newton-Raphson para calcular a raíz quadrada aproximada de um número).

A entrega ocorrerá através de envio dos exercícios compactados no formato `.zip` para a atividade correspondente no Google Classroom, até a data limite da atividade conforme instruções.

Implementações de alunos que contiverem conteúdo simular serão questionadas, podendo haver arguição dos envolvidos. Em caso de confirmação de **plágio** os exercícios copiados de todos os envolvidos, incluindo o autor original, valerão zero.

Sumário

1 Raíz Quadrada Por Aproximação

Exercício 1 (`sqrt.c`). *Uma das formas de se calcular a raíz quadrada de um número real é por meio de aproximações. Uma destas maneiras deriva diretamente da aplicação do método de Newton-Raphson sobre uma função $f(x) = x^2 - a$, onde a é o valor que se deseja calcular $\sqrt{a}$. Assim o problema se transforma em um problema numérico cujo objetivo é calcular uma raíz real para $f(x)$.*

Aplicando as devidas manipulações algébricas chega-se na seguinte equação de recorrência:

$$x_{k+1} = \left(x_k + \frac{a}{x_k}\right) \times 0.5 \tag{1}$$

Ao aplicarmos a Equação 1 sucessivamente a partir de um valor inicial x_0 informado gera-se uma sequência de resultados $x_1, x_2, x_3, \ldots$ que convergem para o valor de $\sqrt{a}$. A cada valor gerado chega-se mais próximo do valor exato da raíz; logo a diferença entre o termo x_i e o termo x_{i+1} da sequência reduz a cada iteração do método. Podemos parar as iterações assim que a diferença entre dois termos consecutivos for menor do que um erro aceitável informado pelo usuário (por exemplo $\epsilon = 0.00001$ para erros com aproximação de 5 casas decimais, $\epsilon = 0.01$ para erros na segunda casa decimal, e assim segue).

Com base no que foi informado, o Algoritmo 1 dado a seguir implementa, em pseudo-código, a lógica por trás do cálculo de $\sqrt{a}$ por aproximação.

Algoritmo 1 RaizQuadAprox(`numero`, `inicial`, `erro`)

1: $x_0 \leftarrow$ `inicial`
2: **repeat**
3: $\quad x_{k+1} \leftarrow \left(x_k + \frac{\texttt{numero}}{x_k}\right) \times 0.5$
4: $\quad$ `diferenca` $\leftarrow$ `abs`$(x_{k+1} - x_k)$
5: **until** (`diferenca` $\leq$ `erro`)
6: **return** x_{k+1}

Neste exercício pede-se que o aluno escreva um programa que implemente o algorítmo de raíz quadrada por aproximação dado acima. O usuário deverá informar qual é o valor de a e o erro aceitável, no formato apresentado acima.

2 Método das Diferenças Finitas

Exercício 2 (`derivada.c`). *Uma derivada de função $f'(x)$ pode ser obtida numericamente utilizando um método denominado de Método das Diferenças Finitas.*

Nela, uma forma simples de estimação de derivada consiste em dados dois pontos do domínio, x e $x + h$, respectivamente, calcula-se uma aproximação da derivada (ou coeficiente da reta tangente) através do cálculo da inclinação da reta secante usando os pontos $\left(x, f(x)\right)$ e $\left(x + h, f(x + h)\right)$ e cuja inclinação é dada pela Equação 2:

$$f'(x) \simeq \frac{f(x+h) - f(x)}{h} \tag{2}$$

onde h é um número suficientemente pequeno que representa uma leve mudança no valor de x, podendo ser positivo ou negativo (por exemplo, $h = 0.0000000000001$).

Isto posto, seja o seguinte problema de física. Sabe-se que uma partícula percorre uma curva obedecendo a seguinte equação de espaço:

$$s = t^2 + t - 2$$

Deseja-se construir uma tabela que mostra, a cada instante de tempo t (i) qual é o espaço percorrido pela partícula, (ii) qual é a velocidade da partícula (obtida por meio da diferenciação numérica da função do espaço), (iii) qual é a aceleração da partícula (obtida pela diferenciação numérica da velocidade).

Assuma que os dados devem ser apresentados em formato tabular, com uma linha para cada instante de tempo, e com quatro colunas (tempo, espaço, velocidade e aceleração). Assuma também que as unidades de medidas estão no sistema internacional (SI), e que as medidas devem ser tomadas de 1 em 1 segundo no intervalo $0 \leq t \leq 60$.

3 Norma de um Vetor

Exercício 3 (`norma.c`)**.** *Seja um vetor v que pertence ao espaço $\mathbb{R}^n$. A norma de um vetor (também chamada de módulo do vetor) é o comprimento deste vetor, calculado pela distância de seu ponto final até a origem. De maneira geral, a norma do vetor mede o tamanho do segmento de reta das coordenadas do vetor v até a origem do sistemas de coordenadas.*

A norma de um vetor com n dimensões pode ser generalizada a partir do Teorema de Pitágoras na fórmula dada na Equação 3, que extrai essa medida a partir da raíz quadrada da soma do quadrado das n componentes do vetor v:

$$\|v\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2} \tag{3}$$

Neste exercício pede-se que o aluno construa um programa que recebe as coordenadas de um vetor $v \in \mathbb{R}^3$, obtenha suas componentes via console, e calcule a norma do vetor correspondente exibindo o resultado no terminal.

4 Calculando Cosseno por Aproximação

Exercício 4 (`cosseno.c`)**.** *Dados dois números $x \in \mathbb{R}$ e $n \in \mathbb{N}$, é possível calcular o valor de $\cos x$ por meio de uma aproximação dada pelos n primeiros termos da série dada pela Equação 4*

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + (-1)^n \frac{x^{2n}}{(2n)!} \tag{4}$$

Escreva um programa que pergunte ao usuário a quantidade de termos n a utilizar na aproximação, e gere uma tabela com valores de cosseno entre 0^o e 90^o usando a aproximação de cosseno dada acima. Lembre-se que:

(i) não existe fatorial em C, logo será preciso criar uma função que faça esse cálculo;

(ii) a aproximação considera o valor x em radianos; logo será preciso converter cada ângulo de graus para radianos antes de expandir a série;

(iii) para cada ângulo dado, deve-se fazer o cálculo novamente pois muda-se o valor de x. Escreva uma função que faça esse cálculo segundo a Equação 4, e chame-a para determinar o cosseno de cada ângulo da tabela.

5 Calculando Logarítmo Natural por Aproximação

Exercício 5 (`logaritmo.c`)**.** *O cálculo do valor do logarítmo natural ou neperiano* $\ln(x)$ *pode ser obtido por meio de aproximação usando a seguinte série de potências (que é convergente no domínio* $0 < x < 2$*) dada na Equação 5:*

$$\ln(x) = (x-1) - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + \dots \tag{5}$$

A qualidade do resultado depende da quantidade de termos usados no somatório. Uma forma de aferir a qualidade da aproximação consiste em computar o ***erro relativo*** E_R *da aproximação que mede o desvio percentual da estimativa* $\overline{e}$ *em relação ao valor esperado e. O* E_R *é calculado por meio da Equação 9:*

$$E_R = \frac{|e - \overline{e}|}{e} \tag{6}$$

Isto posto, escreva um programa em linguagem C que receba como entrada (i) o valor de x cujo resultado deve ser calculado (verifique as condições de convergência); (ii) a quantidade de termos a ser usada no somatório; e que devolva como resultado (i) o valor do $\ln(x)$ *calculado por aproximação e (ii) o erro relativo obtido considerando a estimativa do resultado calculado e o valor obtido pelo uso da função* `log()` *da biblioteca padrão da linguagem C.*

6 Sequência de Fibonacci

Exercício 6 (`fibonacci.c`)**.** *A sequência de Fibonacci é um padrão numérico que ocorre em muitos seres vivos, como é o caso do Nautilus (****Nautilidae****), no fruto de alguns pinheiros, nas pétalas de algumas flores, em girassóis, abacaxis, dentre outros*[1]*.*

[1] `http://britton.disted.camosun.bc.ca/fibslide/jbfibslide.htm`

Foi percebida pela primeira vez no século 12 por Leonardo de Pisa, conhecido como Leonardo Fibonacci. É uma sequência infinita que inicia-se com 0 e 1, que são os dois primeiros termos da sequência. A partir destes termos, para $n \geq 2$, os próximos valores podem ser calculados utilizando-se a relação de recorrência: $F_n = F_{n-1} + F_{n-2}$. Podemos dizer que de forma geral F_n calcula o n-ésimo termo da sequência segundo a Equação 7:

$$F_n = \begin{cases} 0 & se\ n = 0 \\ 1 & se\ n = 1 \\ F_{n-1} + F_{n-2} & se\ n \geq 2 \end{cases} \tag{7}$$

Aplicando a fórmula dada, pode-se gerar os termos da sequência

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \ldots$$

Neste exercício pede-se que o aluno crie um programa capaz de exibir a sequência de Fibonacci até o n-ésimo termo, informado pelo usuário. O programa deverá imprimir um termo da sequência por linha.

Dica: *crie uma função que calcule o valor da sequência para o k-ésimo termo informado, e chame-a para cada termo $1 \leq k \leq n$.*

7 Cálculando a Razão Áurea por Fibonacci

Exercício 7 (`razaoaurea.c`)**.** *No Exercício 6 vimos que é possível gerar os termos da sequência de Fibonacci através da Equação 7. Uma curiosidade matemática existente sobre os termos desta sequência está relacionada à chamada* ***razão áurea*** *ou* ***número de ouro****. Trata-se de um número irracional que pode ser obtido pela Equação 8:*

$$\phi = \frac{1 + \sqrt{5}}{2} \approx 1,618033988749894848\ldots \tag{8}$$

Desde muito tempo a razão área é utilizada nas artes, em especial quando o artista buscava representar a 'perfeição da beleza'. Obras como 'O Nascimento de Vênus' (Botticelli), 'O Sacramento da Última Ceia' (Salvador Dalí), 'Mona Lisa' (Da Vinci), nas pinturas do Mestre Giotto, dentre outras obras de artistas renascentistas possuem elementos cujas medidas são expressas pela razão áurea. Também se observa a ocorrência da razão áurea na natureza, como por exemplo no tamanho das falanges da mão humana, nas colmeias, na concha do nautilus, na proporção entre abelhas machos e fêmeas, em crescimento de plantas e em outros aspectos ligados a ordem de crescimento da natureza.

Um valor aproximado para o número irracional ϕ pode ser obtido pela razão entre o termo F_n pelo termo antecessor F_{n-1} da sequência de Fibonacci para $n \geq 2$, isto é:

$$\phi \approx \frac{F_n}{F_{n-1}}$$

Quanto maior o termo n usado na aproximação, mais próximo é o resultado obtido do número ϕ calculado pela Equação 8. Para medir a diferença entre o número aproximado $\overline{x}$ de um número alvo

x pode-se utilizar uma medida de distância denominada 'erro absoluto', de cálculo simples e cujo valor pode ser obtido pela Equação 9:

$$EA_{\overline{x}} = |\overline{x} - x| \tag{9}$$

Quanto menor for o valor de $EA_{\overline{x}}$, mais próximo está a estimativa do valor exato.

Neste exercício pede-se que o aluno construa um programa em C que recebe como entrada um valor de erro que deve ser atingido no cálculo da estimação de ϕ pela razão entre F_n e F_{n-1}. Este valor deverá estar no seguinte formato: 0.1 *para erro na primeira casa decimal,* 0.01 *para erro na segunda casa decimal,* 0.001 *para erro na terceira casa decimal, e assim segue. O programa deverá calcular a razão áurea aproximada, medir a distância entre o valor exato (dado pela Equação 8) e o valor aproximado, e encerrar o programa se o erro absoluto calculado for menor que o erro estimado. Caso não seja, repetir o processo, mas considerando o próximo termo da sequência de Fibonacci, até que o erro absoluto seja menor do que o erro informado pelo usuário.*

Dica: *crie uma função que devolve o valor exato de ϕ, crie uma função que calcule o n-ésimo termo da sequência de Fibonacci, e faça os cálculos iniciando por $^{F_2}/_{F_1}$ e aumentando iterativamente o valor de n até que a medida de erro seja menor que o informado pelo usuário.*

8 Representação de Polinômio

Exercício 8 (`polinomio.c`)**.** *Escreva um programa em linguagem C que seja capaz de representar uma TAD de um monômio, com coeficiente e grau do monômio. Considere que uma listagem de monômios é um polinômio (por exemplo, $p(x) = 2x^3 + 0.5x^2 - 3x + 2$ é um polinômio formado pelos monômios $2x^3$, $0.5x^2$, $3x^1$, $2x^0$).*

Uma sugestão de representação é usar uma `struct` *para representar o monômio, e um vetor de monômios para representar o polinômio além de outras variáveis de controle, como por exemplo, o total de monômios do polinômio.*

De posse da representação o programa deverá perguntar ao usuário quantos monômios deve-se usar, e perguntar qual o coeficiente e grau de cada um. Em seguida o programa deverá perguntar ao usuário se deseja avaliar o polinômio em questão (isto é, para um dado valor de x informado pelo usuário, calcular o valor de $p(x)$ correspondente). Após avaliar o programa deverá perguntar se deseja avaliar outro valor de x, ou encerrar a execução do mesmo.

9 Derivada de Polinômios

Exercício 9 (`derivada.c`)**.** *Neste exercício deve-se escrever um programa em linguagem C que receba como entrada a quantidade de monômios de um polinômio $p(x)$, seus coeficientes e graus, e que construa um novo polinômio $p'(x)$ contendo a derivada do polinômio informado.*

De posse do polinômio $p'(x)$ o programa pergunta um valor de x para calcular o valor da derivada no ponto x. Para isso deve-se avaliar o valor que $p'(x)$ possui no valor x informado. Após calcular e informar o resultado, o programa deverá perguntar ao usuário se deseja calcular $p'(x)$ para um

novo x; em caso positivo deverá repetir o processo questionando novo x e devolvendo ao usuário $p'(x)$. ***Dica****: utilize a TAD definida no Exercício 8.*

10 Integral de Polinômio

Exercício 10 (`integral.c`)**.** *Neste exercício deve-se escrever um programa em linguagem C que receba como entrada a quantidade de monômios de um polinômio $p(x)$, seus coeficientes e graus, e que construa um novo polinômio $P(x)$ contendo a primitiva (ou anti-derivada) do polinômio informado (sem considerar a constante c) tal que $P'(x) = p(x)$.*

De posse do polinômio $P(x)$ o programa pergunta os limites do intervalo de integração para a integral definida $\int_a^b p(x)\,dx$, e efetua a diferença $P(a) - P(b)$ que representa a área sobre a curva do polinômio no intervalo $[a, b]$. Para efetuar a diferença, o polinômio $P(x)$ necessita ser avaliado para $x = a$, e para $x = b$ informados. ***Dica****: utilize a TAD definida no Exercício 8.*

11 Média e Variância Amostral

Exercício 11 (`media-var.c`)**.** *Neste exercício o programador deverá construir um programa em linguagem C que recebe uma lista de números reais de tamanho variado, informado pelo usuário, que representam valores observados de algum fenômeno. Na notação matemática a seguir n indica o número de observações tomadas, enquanto que x_i indica o valor da i-ésima observação.*

Após receber estes dados o programa deverá calcular a média amostral, cujo cálculo é dado pela Equação 10:

$$\overline{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{10}$$

De posse da média, o programa deverá calcular também a variância amostral, cujo cômputo é feito mediante aplicação da Equação 11:

$$s^2 = \frac{\sum_{i=1}^{n} \left(x_i - \overline{x}\right)^2}{n-1} \tag{11}$$

Por fim, média e variância amostrais são apresentadas como resultado no console.

12 Ordenação de Vetores

Exercício 12 (`ordenacao.c`)**.** *Um algoritmo de ordenação consiste em um método computacional que recebe um vetor V de dimensão n contendo valores numéricos desordenados, e retorna o vetor V ajustado com os elementos menores precedendo os maiores (isto é, coloca-os em ordem).*

Um dos algoritmos mais famosos de ordenação de fácil implementação é o algoritmo conhecido por 'Ordenação por Seleção'. Ele recebe um vetor V de entrada e varre-o em ordem crescente de

índice, buscando um substituto de menor magnitude dentre os elementos ainda não inspecionados para substituir pelo elemento da i-ésima posição em inspeção. Um pseudo-código da ordenação por seleção é dada no Algoritmo 2.

Algoritmo 2 `ORDENACAO-POR-SELECAO`(V: vetor, n: inteiro)

```
1: for (i ← 1 to n) do
2:    pos_menor ← i
3:    for (j ← i + 1 to n) do
4:       if (V[j] < V[pos_menor]) then
5:          pos_menor ← j
6:       end if
7:    end for
8:    aux ← V[i]
9:    V[i] ← V[pos_menor]
10:   V[pos_menor] ← aux
11: end for
```

Neste exercício pede-se que o programador construa uma aplicação em linguagem C que receba um conjunto de números inteiros de tamanho variável, informados pelo usuário, e recebidos em qualquer ordem. Após receber todas as entradas aplicação deverá ordenar estes elementos em ordem crescente de magnitude. Ao encerrar a ordenação deve-se imprimir a lista de números na tela, separados pelo caracter vírgula (`,`).

13 Cálculo da Mediana e Moda

Exercício 13 (`mediana-moda.c`)**.** *Em estatística, a* ***mediana*** *de uma amostra é uma medida de tendência central que divide os dados em duas partes iguais, metade abaixo da mediana, e metade acima da mediana. Já a* ***moda*** *é a observação de ocorrência mais frequente da amostra.*

Neste exercício pede-se que se construa um programa em linguagem C que receba uma lista com números inteiros de tamanho variável, informado pelo usuário, que representam observações tomadas de um dado fenômeno.

O programa deverá calcular qual é a mediana e a moda da amostra observada. Para tal, deve-se ordenar a amostra de valores (sugere-se aproveitar o algoritmo descrito no Exercício 12), calcular a posição que corresponde ao meio do vetor e devolver o elemento posicionado nesta posição após ordenação. Para a moda é preciso computar a quantidade de vezes que cada valor observado ocorreu na amostra, e devolver o mais frequente.

Dica*: faça uma varredura no vetor de valores observados, tome um por vez, e refaça a varredura contando quantos elementos de posições distintas correspondem à ele. Repita o processo para cada elemento do vetor, e devolva o mais frequente.*

14 Produto Matricial

Exercício 14 (`prodmatriz.c`). *O Algoritmo 3 descreve, em pseudo-código, as operações necessárias para realizar o produto de uma matriz A_{nn} por uma matriz B_{nn}.*

Algoritmo 3 `OPERACAO-COM-MATRIZES`(A: matriz $n \times n$, B: matriz $n \times n$)

```
1: {Realiza a multiplicação C = A × B}
2: for i ← 1 to n do
3:    for j ← 1 to n do
4:       c_ij ← 0
5:       for k ← 1 to n do
6:          c_ij ← c_ij + a_ik · b_kj
7:       end for
8:    end for
9: end for
```

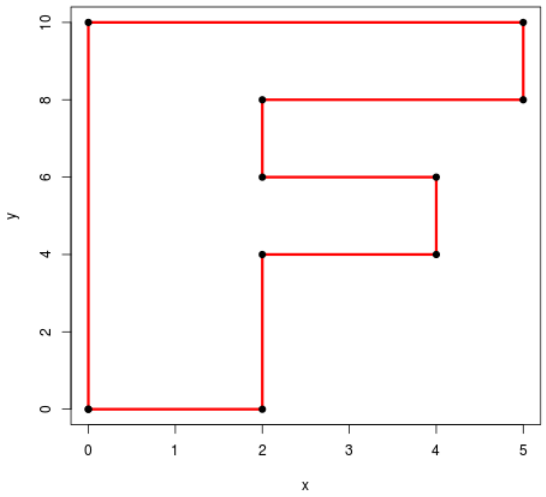
Isto posto, escreva um programa em linguagem C que receba as dimensões de duas matrizes quadradas A e B, preenche cada elemento das matrizes segundo o que o usuário informar via console aplica o Algoritmo 3 para calcular o produto matricial $A \times B$.

15 Rotação 3D

Exercício 15 (`rotacao3d.c`). *Objetos são representados em ambientes virtuais por meio de um conjunto de faces, cada qual formada por um conjunto de vértices ou pontos no espaço 3D.*

Seja um objeto em forma da letra 'F' composto pelos seguintes pontos no espaço 3D (acompanhe no gráfico abaixo cada ponto da tabela com cada vértice da face que desenha a letra 'F' no plot gráfico):

Pto	*Coord. X*	*Coord Y*	*Coord. Z*
p_1	0	0	0
p_2	0	10	0
p_3	5	10	0
p_4	5	8	0
p_5	2	8	0
p_6	2	6	0
p_7	4	6	0
p_8	4	4	0
p_9	2	4	0
p_{10}	2	0	0
p_{11}	0	0	0



*Quando um objeto é **rotacionado** no mundo virtual, ocorre uma rotação de seus pontos ao redor de um eixos de rotação. Para cada eixo existe um ângulo θ associado com a rotação esperada. No caso*

da rotação no eixo Z (isto é, quando gira-se o objeto no plano XY) a matriz de rotação associada é a que segue:

$$\begin{bmatrix} x'_i \\ y'_i \\ z'_i \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\textit{matriz de rotação em } Z} \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Neste exercício pede-se que o aluno implemente uma aplicação que recebe como entrada o valor do ângulo de rotação no eixo Z em graus (dado por θ), converta-o em radianos, construa a matriz de rotação correspondente e implemente o produto matricial dado acima para transformar cada ponto (x_i, y_i, z_i) da figura original em um ponto (x'_i, y'_i, z'_i) da figura rotacionada.

***Dica**: Use as funções de seno e cosseno da biblioteca `math.h`, e adapte o algoritmo do Exercício 14 para operar sobre uma matriz A de dimensões $n \times n$ e vetor de entrada V de dimensões $n \times 1$ (observe que neste caso o produto Av resulta em v' de dimensões $n \times 1$).*

Referências

[Ascencio] ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. Fundamentos da Programação de Computadores: Algoritmos, Pascal, C/C++ e Java. 2a ed. São Paulo:, Pearson Education, 2008.

[Deitel] DEITEL, Paul J.; DEITEL, Harvey M. C: Como programar. 6a ed. São Paulo: Prentice Hall, 2013.

[Kernighan] KERNIGHAN, Brian W.; RITCHIE, Dennis M. C: a linguagem de programação padrão ANSI. Rio de Janeiro: Elsevier, 1990.

[Celes] CELES, Waldemar. CERQUEIRA, Renato. RANGEL, José Lucas. Introdução a Estrutura de Dados: com técnicas de programação em C. Rio de Janeiro: Elsevier, 2004.

[Forbellone] FORBELLONE, A. L.; EBERSPACHER, H. Lógica de Programação. 3a ed. São Paulo: Editora Pearson Prentice-Hall, 2005 [recurso eletrônico].

[Mizrahi (1994)] MIZRAHI, V. V. Treinamento em Linguagem C - Módulo 1. São Paulo: Pearson Makron Books, 1994 [recurso eletrônico].

[Mizrahi (2001)] MIZRAHI, V. V. Treinamento em Linguagem C - Módulo 2. São Paulo: Pearson Makron Books, 2001 [recurso eletrônico].

[Ziviani] ZIVIANI, Nívio. Projeto de Algoritmos: com implementação em Pascal e C. 3a ed. revista e ampliada. São Paulo: Cengage Learning, 2011.