Gilvan Vilarim

Algoritmos

Programação Para Iniciantes

3ª edição

EDITORA
CIÊNCIA MODERNA

Algoritmos: Programação para Iniciantes – 3ª Edição

Editor: Paulo André P. Marques
Produção Editorial: Dilene Sandes Pessanha
Capa: Marcia Lips
Diagramação: Gilvan de Oliveira Vilarim
Copidesque: Daniela Marrocos

FICHA CATALOGRÁFICA

V697a
Vilarim, Gilvan de Oliveira.

Algoritmos: programação para iniciantes – 3ª edição.
Rio de Janeiro: Editora Ciência Moderna Ltda., 2017.

304 p.: il.
Inclui índice e bibliografia

1. Informática. 2. Programação de computador. I. Título.
ISBN: 978-85-399-0835-6

CDD 001.642
005

Editora Ciência Moderna Ltda.
R. Alice Figueiredo, 46 – Riachuelo
Rio de Janeiro, RJ – Brasil – CEP: 20950-150
Tel: (21) 2201-6662/ Fax: (21) 2201-6896
E-mail: lcm@lcm.com.br
www.lcm.com.br 01/17

Agradecimentos

O apoio da Fundação Educacional Serra dos Órgãos (FESO) foi fundamental para este trabalho. Nela nasceram as primeiras ideias deste projeto, em calorosas discussões na época da mudança da grade curricular do curso de Ciência da Computação. Na 3ª edição, tive apoio do Instituto Federal do Rio de Janeiro (IFRJ).

O professor e colega João Fernando Diniz Falcão deu valiosas sugestões em rascunhos do livro e, desde o início, incentivou a sua elaboração. Agradeço pela atenção e disponibilidade para ler esses rascunhos.

Ensino é troca. Agradeço a todos os alunos com os quais convivi durante todos estes anos; aprendi muito com vocês.

Desde a primeira edição, sugestões foram dadas por pessoas de várias partes do Brasil. Agradeço a cada uma delas pelas contribuições.

Por último, faço um agradecimento especial à minha esposa Márcia, que sempre me apoiou e soube compreender o sacrifício do nosso tempo para a realização deste projeto pessoal.

Este livro é para Márcia S. Guerreiro.

Prefácio

A ideia de escrever a primeira edição deste livro surgiu no final de 2001, entre reuniões, conversas e cafezinhos com outros colegas professores. Discutíamos que o Brasil possuía um mercado restrito de livros sobre introdução à programação de computadores, concentrado no material oferecido por poucos autores nacionais. Pensou-se, então: por que não uma alternativa? Propus-me a escrever um livro que abordasse o mesmo tema, com um texto totalmente novo e agregando algumas inovações, pensadas com base na minha experiência com o ensino de programação. Algumas dessas inovações seriam:

- Novo português estruturado, que simplifica os detalhes das linguagens de programação, mas também procura acostumar o leitor aos formalismos da escrita, necessários para o aprendizado de linguagens reais;
- Exercícios de fixação e problemas propostos, que praticam insistentemente a mecânica de certas estruturas de programação e exigem raciocínio para a resolução de problemas;
- Inclusão de curiosidades sobre computadores, algoritmos e programação, que despertam e estimulam o interesse pelo tema;
- Inclusão de respostas para as perguntas mais comuns feitas pelos iniciantes em programação.

Foi a partir daí que surgiu este material, cuja base maior foram anotações e discussões feitas em aulas de programação. Boa parte da tônica do texto é um espelho de apresentações e discussões ocorridas na própria sala de aula; procurei a todo instante colocar-me no papel do aluno e considerar suas dúvidas, curiosidades e dificuldades iniciais.

Mais de 10 anos após o seu lançamento, chegamos à nossa 3ª edição! Ao longo do tempo, a primeira e segunda edições foram adotadas como bibliografia em diversos cursos no Brasil. Passamos pela revitalização da rede de educação profissional e tecnológica, que disseminou nosso texto em diversos cursos de níveis médio e superior. Mais recentemente, foi reacesa a discussão da importância do pensamento computacional em escolas; isso mostra que você pode aprender a programar sem a intenção de ser um profissional da Computação, pois o raciocínio lógico trazido pela programação é benéfico para muitas outras áreas.

Atualmente, temos uma adoção distribuída entre cursos de graduação e cursos de nível técnico, fora profissionais e outros interessados que adquiriram o livro por vontade de conhecer os fundamentos da programação. Tivemos um retorno positivo de diversas regiões do Brasil, com opiniões e sugestões de melhoria, e isso fortaleceu a necessidade de se fazer uma boa revisão no texto original.

Como Usar Este Livro

Se você está iniciando o estudo de programação de computadores, este livro não deve ser "consumido" em um curto espaço de tempo. Procure ler os capítulos com calma, avançando apenas quando estiver seguro dos conceitos estudados. Os exercícios apresentados são uma forma de praticar o método para a resolução de problemas e servem para fixar a sintaxe apresentada para a construção das soluções.

Cada exemplo apresentado é estratégico para o entendimento dos assuntos, pois aborda os conceitos básicos e serve para a compreensão de situações mais complexas. Acompanhe cada um cuidadosamente.

Os exercícios de fixação, como o próprio nome indica, são colocados para aumentar o grau de retenção de cada assunto. Fazê-los é uma forma de se obrigar a reler o texto e praticar a escrita dos algoritmos.

Os problemas propostos vão além da mera fixação e procuram desenvolver também o raciocínio para a resolução de problemas. No nível de aprendizagem proposto, cada problema é uma simplificação de situações que um programador pode encontrar no mundo real, onde a própria compreensão do que deve ser feito é parte do método de solução. São também propostos alguns algoritmos comuns aos estudos de programação.

Todos os exercícios marcados com o sinal § possuem uma solução incluída em um dos anexos do livro. Como você verá, é possível construir diferentes soluções corretas para um mesmo algoritmo. Para vários casos, considere que a solução apresentada é uma das muitas que são possíveis.

Você não precisa ter um computador para usar este livro. Como verá adiante, a construção de algoritmos antecede a programação em si, de modo que lápis e papel são suficientes para praticar as atividades.

A Quem Se Destina Este Livro

Este livro foi feito para quem deseja ter o primeiro contato com a programação de computadores, em particular, a construção de algoritmos. Há um grande esforço em fazer com que este seja seu **primeiro** livro de programação, procurando despertar-lhe o gosto por esta fascinante área da Ciência da Computação. Embora possa ser usado por autodidatas, nossa intenção é que este material seja um complemento às aulas de cursos introdutórios de programação.

O foco neste tipo de público exclui, portanto, a apresentação de outros conteúdos mais avançados. Não pretendemos, por exemplo, apresentar uma linguagem por completo ou explorar determinadas técnicas de programação, mas sim, mostrar e exercitar os princípios que regem a construção de bons algoritmos, procurando, ao máximo, criar hábitos saudáveis para o aprendizado de outros assuntos relacionados à programação.

Nada impede, ainda, que você já saiba programar e queira utilizar o livro para reforçar conceitos e praticar exercícios. Para isto, incluímos um anexo mostrando a conversão de nossos algoritmos para a sintaxe de três linguagens usadas no ensino de programação: Pascal, C e Python. Outra opção é ler o livro de forma concomitante ao estudo dessas linguagens, ou mesmo outras, de caráter introdutório.

Aos Professores

Acreditamos que o professor dos fundamentos de programação tem uma tarefa hercúlea: cativar e preparar os alunos em um dos assuntos mais importantes da computação, estimulando um novo jeito de pensar que, para muitos, é uma grande novidade. O aprendizado, portanto, passa a ser um compromisso mútuo entre aluno e professor.

Este livro foi escrito fortemente baseado na vivência com o ensino de programação. Seu texto procura transferir para o papel as explicações e os exemplos usados em aulas de algoritmos e programação. Tanto no Brasil como em outros países, ainda não há um consenso sobre o melhor modo de iniciar o aluno na programação de computadores. Nossa opção foi desmembrar o estudo de algoritmos das linguagens específicas e das arquiteturas de computadores, enfocando muito mais a **resolução de pro-**

blemas; nosso esboço de computador é uma abstração (bastante) livre de uma máquina de von Neumann.

Para a montagem da solução, usamos uma linguagem estruturada, muito inspirada na linguagem Pascal, mas com liberdade para algumas mudanças que facilitassem o aprendizado (uma delas, por exemplo, foi preferir usar apenas operadores aritméticos simbólicos, retirando DIV e MOD, que causam confusão com os comandos e funções).

Algumas consequências destas opções foram:

- Uma grande preocupação em mostrar e "mastigar" com bastante calma os conceitos iniciais de cada assunto;
- Resolução e comentários de exercícios considerados como base para a resolução de problemas mais complexos, tentando "pensar em voz alta" com o leitor como se atinge cada solução;
- Carga de exercícios de fixação para praticar conceitos, comandos e sintaxes, e complementação com exercícios propostos, que problematizam situações e exigem tanto a compreensão e identificação do problema quanto a montagem de sua solução;
- Seção com perguntas e respostas, incluindo explicações para perguntas normalmente feitas por alunos que começam a estudar algoritmos.

Não se teve a intenção, todavia, de esgotar a teoria e a prática dos assuntos, de modo que cada tópico pode ser complementado de acordo com o ritmo e a carga das aulas. Outros materiais, portanto, podem ser indicados à parte.

Estrutura do Livro

Nosso material é dividido em grandes tópicos, comuns no estudo de algoritmos. Além disso, colocamos anexos que podem ser úteis para o leitor complementar os seus conhecimentos.

O Capítulo 1 apresenta um breve histórico da evolução dos computadores, preparando o aluno com conceitos que convergem para o estudo de programação. Um modelo virtual de um computador é apresentado, servindo de base para promover a construção de algoritmos.

O Capítulo 2 apresenta o ferramental usado para montar os nossos algoritmos. Mostramos como os algoritmos podem ser representados; apresentamos noções fundamentais, como as do tipo de dado e variável; indicamos como o computador se comunica com o meio externo; e finalizamos com o estudo de um exemplo simples.

O Capítulo 3 já põe em prática diversos conceitos, procurando exercitar a lógica de programação com algoritmos sequenciais. Apresentamos também uma forma de dividir o algoritmo em etapas menores, o que facilita a montagem da solução.

Os Capítulos 4 e 5 agregam duas outras estruturas básicas aos algoritmos sequenciais: testes e repetições. O Capítulo 4 mostra como podemos ensinar o computador a tomar decisões, executando ou não determinadas ações a partir de condições lógicas. Já o Capítulo 5 mostra como pedir à máquina que ela repita determinadas ações em um algoritmo.

Os Capítulos 6 e 7 apresentam tipos de dados estruturados. Com eles, podemos construir algoritmos mais complexos, capazes de manipular os dados na memória com estruturas baseadas em tipos mais simples.

O Capítulo 8 consolida os conceitos vistos nos capítulos anteriores sob o tema Programação Estruturada, alinhavando os assuntos e formalizando as boas práticas de programação. O último capítulo tece comentários finais e apresenta sugestões de estudos futuros.

O livro possui alguns anexos incluídos ao seu final. O primeiro apresenta brevemente a estrutura das linguagens de programação populares – Pascal, C e Python – para aqueles que desejarem implementar os algoritmos estudados. O segundo anexo apresenta alguns recursos relacionados à independência entre os módulos, complementando algumas questões vistas ao longo do livro. O terceiro anexo mostra outra forma de representação de algoritmos, mediante o uso de fluxogramas. O quarto anexo mostra uma notação formal para descrever a sintaxe dos comandos usados para construir os algoritmos. O último anexo apresenta respostas para os exercícios de fixação, bem como a resolução de diversos problemas propostos.

Novidades da 3ª edição

- O texto passou por uma ampla revisão. Várias coisinhas foram modificadas e alguns conceitos foram clarificados;
- Os algoritmos foram adaptados para a prática com uma sintaxe mais parecida com as linguagens de programação populares;
- Os problemas apresentados foram revistos, com a exclusão, inclusão e modificação de vários exercícios;
- Para conhecer as linguagens de programação reais, o anexo já existente, que antes apresentava um pouco da linguagem Pascal, agora também mostra conceitos de C e Python;
- Um novo anexo explora um pouco mais as questões relacionadas à modularização de algoritmos, analisando a importância da independência entre os módulos.

O Autor

Gilvan Vilarim é bacharel em Informática pelo IME-UERJ, mestre em Engenharia de Sistemas e Computação pela COPPE-UFRJ e doutor em Serviço Social pela ESS-UFRJ. Leciona desde 1995, atuando principalmente nos ensinos técnico e de graduação na área de computação; trabalhou em diversas instituições públicas e particulares ao longo dos anos. Atualmente é professor do Instituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro (IFRJ).

Para um currículo mais detalhado, procure-o na Plataforma Lattes, disponível na Internet:

http://lattes.cnpq.br/3537226826949396

Sumário

Capítulo 3. Construção de Algoritmos

Os objetivos deste capítulo são:

- Formalizar um método para a verificação manual do comportamento de um algoritmo;
- Mostrar como podemos dividir problemas maiores em subproblemas, facilitando a construção de soluções;
- Praticar a construção de algoritmos simples em português estruturado.

3.1. Roteiro para a Construção de Algoritmos

Mesmo com todos os conceitos vistos até agora, a construção de um algoritmo parte sempre de algo fundamental: o problema. A montagem da solução depende da prática e da criatividade do programador, que podem ser conseguidas exercitando esses conceitos. Para um mesmo problema, programadores diferentes podem ter soluções diferentes e corretas.

As etapas principais para a construção de um algoritmo são:

1. **ENTENDER O PROBLEMA**. Acredite, esta é a etapa **mais importante** na programação de computadores. Você deve ser capaz de compreender o que está sendo solicitado; se não entender, poderá construir uma solução errada ou até construir uma solução correta para o problema errado!

2. Identificar no problema as **SAÍDAS** do algoritmo, pois são aonde se deseja chegar. Atente para o que deve ser calculado, impresso, processado.

3. Identificar no problema as **ENTRADAS** do algoritmo, isto é, quais dados a máquina necessita que o usuário forneça para fazer os cálculos. Atente para o que deve ser lido, fornecido à máquina, obtido do usuário.

4. Atentar para o **PROCESSAMENTO** em si. Identifique quais cálculos são necessários, sempre a partir dos dados de entrada para chegar aos dados de saída. Rascunhe os cálculos, considerando que as operações possíveis serão aquelas disponíveis no português estruturado.

5. Tentar separar, na medida do possível, as **ETAPAS** de entrada, processamento e saída dentro do algoritmo.

6. Colocar os comandos no algoritmo conforme uma **SEQUÊNCIA** básica. Lembre-se, acima de tudo, que a máquina executa o algoritmo seguindo exatamente a sequência definida por você.

Ao analisar um problema, muitas vezes você precisa partir de textos. Uma dica é ficar de olhos nas ações apresentadas nos problemas, ou seja, os **verbos**. Verbos como ler, obter, receber, pegar, entre outros, dão normalmente uma ideia de entrada de dados. Verbos como exibir, mostrar, imprimir, escrever, apresentar, entre vários outros, passam uma ideia de saída de dados – sempre tendo como referencial o computador. Os demais verbos também podem fazer você chegar à conclusão de que há um processamento a ser feito: calcular, transformar, converter, manipular. E é claro que existem ações fazendo apenas "parte do enredo", sem reflexo direto na solução.

3.2. Verificação Manual de Algoritmos

Após a construção de um algoritmo, existe uma forma "braçal" de verificar se ele está realmente apresentando uma solução correta para o problema. Essa forma, chamada de "**método chinês**" (ou ainda "teste de mesa", ou mesmo "chinezinho"), consiste em simular manualmente a execução do algoritmo, comando após comando, atentando para as entradas e as saídas de dados, e principalmente acompanhando o comportamento das variáveis que foram utilizadas.

"Fazer o chinês", como também se diz no jargão de computação, equivale a se colocar no papel do computador, anotando o que a máquina faria ao encontrar cada comando.

1º EXEMPLO

Calcular a média de dois números reais fornecidos pelo usuário.

SOLUÇÃO

Rascunhando uma solução para o problema, podemos identificar:

Entradas: dois números reais;

Saída: a média dos números;

Cálculo necessário: somar os números e dividir o resultado por 2 (quando não há uma menção diferente, supõe-se que seja um cálculo de média aritmética, pois é o mais comum).

A solução deste problema pode tomar como base o algoritmo para somar dois números, acrescentando uma variável:

```
// Algoritmo para calcular a média de 2 números
Variáveis
   num1, num2, soma, media :real;

Início
   ler num1;
   ler num2;
   soma <- num1 + num2;
   media <- soma / 2;
   escrever media;
Fim.
```

Como saber se o algoritmo está realmente calculando o valor correto? Inicialmente, anotamos as variáveis do algoritmo e, ao lado de cada uma, o valor nelas armazenado (o valor inicial é indefinido e, por isso, colocamos o sinal de interrogação na anotação).

```
"Chinês"
num1 ?
num2 ?
soma ?
media ?
```

Na primeira leitura, suponha que o usuário forneça o valor 8. Neste caso, as variáveis ficam com a seguinte situação na memória:

```
"Chinês"
num1 8
```

```
num2 ?
soma ?
media ?
```

Na segunda leitura, suponha que o usuário forneça, agora, o valor 7. As variáveis ficam com a seguinte situação:

```
“Chinês”
num1 8
num2 7
soma ?
media ?
```

Na sequência, o computador calcula a soma dos números:

```
“Chinês”
num1 8
num2 7
soma 15
media ?
```

E depois, calcula a média na atribuição seguinte:

```
“Chinês”
num1 8
num2 7
soma 15
media 7.5
```

O valor 7.5 será impresso em um dispositivo de saída. De fato, a média dos números 8 e 7 é 7.5, e o algoritmo fez o cálculo corretamente. Por uma questão de bom senso, podemos colocar valores de entrada que facilitem os cálculos e que nos permitam antecipar a resposta esperada (como em uma prova real). Se fôssemos supor os valores de entrada 6.35 e 4.92, teríamos muito mais trabalho, certo?

2º EXEMPLO

Indicar os valores impressos pelo algoritmo a seguir:

```
// Algoritmo para praticar o método chinês
Variáveis
   a, b, c, d :inteiro;

Início
   a <- 10;
   b <- 20;
   d <- a + a;
   c <- a + b;
   a <- b + c;
   d <- d + 1;
   escrever a;
   escrever c;
   escrever d;
Fim.
```

SOLUÇÃO

A única utilidade deste algoritmo é servir como prática do "chinês". Veja a variação do conteúdo das variáveis:

```
"Chinês"
a ? 10{1} 50{5}
b ? 20{2}
c ? 30{4}
d ? 20{3} 21{6}
```

Os números entre chaves não são necessários; eles foram colocados para entender em qual ordem os valores foram armazenados nas variáveis, pela sequência do algoritmo. Repare que algumas variáveis foram utilizadas mais de uma vez e, por conceito, o valor anterior nelas armazenado é perdido para dar lugar ao novo. Uma forma de não nos perdemos nesses valores é riscar o valor anterior da variável, imediatamente quando ela recebe um novo valor. Os últimos valores armazenados – 50, 30 e 21 – ficam mais para a direita e serão aqueles impressos no dispositivo de saída, nesta ordem.

Observe que a anotação das variáveis não representa necessariamente a ordem de exibição; isso depende da ordem de colocação dos comandos de saída. A variável `b` nem foi impressa, apesar de ter sido utilizada internamente no algoritmo.

3º EXEMPLO

Ler o preço de um par de sapatos em uma loja e escrevê-lo com um desconto de XX%.

SOLUÇÃO

O algoritmo deve exibir como resposta o preço de um par de sapatos já com um desconto, de modo que já precisamos de uma variável para armazená-lo. Para este cálculo, o usuário deve fornecer o preço original dos sapatos e também o valor do desconto (no enunciado, XX indica algo que varia a cada execução do algoritmo).

Temos, então, um algoritmo com três variáveis: duas de entrada e uma de saída. Mas, como calcular o preço com desconto? Vamos rascunhar primeiro um exemplo numérico, no qual o par custa 50 reais e o desconto é de 20%:

Valor a descontar = 20% de 50 reais = 50 * 20/100 = 10 reais

O valor de 20% é lido "vinte por cento", ou seja, 20 dividido por 100. Ao ser multiplicado pelo preço, teremos o valor do desconto. Logo, o valor com desconto será:

50 - 10 = 40 reais

Entretanto, se o preço original estiver em uma variável `preco` e o desconto for guardado na variável `valorADescontar`, poderemos montar uma expressão mais genérica para o cálculo:

valorADescontar = preco * desconto / 100

precoFinal = preco – valorADescontar

Temos, então, o algoritmo a seguir:

```
// Algoritmo para calcular o desconto do preço do sapato
Variáveis
   preco, desconto, valorADescontar, precoFinal :real;
```