2. Algoritmo e suas Representações

Neste capítulo você aprenderá a:

- Identificar um algoritmo no dia a dia.
- Diferenciar entre os tipos de representação de algoritmos.
- Aplicar algoritmos na resolução de problemas.

2.1 Algoritmo

Na minha interpretação, o algoritmo pode ser entendido como um molde para o que virá a se tornar um programa, ou seja, é um elemento que possui todos os itens necessários para desenvolvermos nosso programa. Ok, talvez esta palavra nova, algoritmo, e o que eu disse possa não fazer muito sentido para você, então vamos a uma analogia.

Pare por alguns minutos e reflita sobre o que você fez em um determinado momento do dia, como, por exemplo, do acordar até tomar café da manhã. No meu caso, posso listar que ocorreram as seguintes ações nesta ordem:

1. Levantei da cama;
2. Coloquei comida para as gatas;
3. Levei os cachorros para o quintal;
4. Arrumei a cama;
5. Levei as gatas para o quintal;
6. Preparei o café;
7. Tomei café.

Observe que os passos realizados estão em uma linguagem que não expressa detalhes do que foi feito em cada uma dessas ações, é o que chamamos de linguagem de *alto-nível*. Embora eu não tenha descrito detalhes do que houve em cada ação, conseguimos identificar uma ordem e o que foi realizado. Esse passo a passo, cronológico, é um exemplo de algoritmo.

Outro fato interessante de um algoritmo é que ele determina uma rotina. A rotina é algo que pode ser replicada, como a nossa rotina matinal. Essa característica do algoritmo é essencial para desenvolvermos aplicações escaláveis (que evoluem), pois não é preciso

começar do zero; podemos reaproveitar rotinas e sub-rotinas, caso necessitemos, realizadas por nós mesmos ou por terceiros. Na pesquisa científica em computação, os autores devem mostrar em seus artigos, quando envolver a produção de um programa (implementação), os algoritmos utilizados. Tendo como objetivo permitir com que outros pesquisadores possam replicar os seus resultados. Isso permite avanço científico.

Ao unir minha interpretação e o exemplo vamos para um conceito de algoritmo mais formal. O algoritmo é uma especificação de uma sequência ordenada de instruções, finitas e não-ambíguas (sem redundância), que deve ser seguida para a solução de um determinado problema, garantindo a sua repetibilidade (JUNIOR, 2009). A partir dessa definição você consegue visualizar um exemplo de algoritmo no seu dia a dia? Vou citar um bem simples: receitas culinárias. Elas mostram passos (ordenados) para se produzir como produto final um prato delicioso (pelo menos é o que esperamos).

Creio que temos agora uma noção melhor do que seja um algoritmo, mas você deve querer saber como o utilizaremos para programação. Na Seção 1.1, falei que temos que deixar a mente se acostumar com uma nova forma de pensar. O algoritmo é essa nova forma de pensar. Suponha que você quer beber um copo com água, como você pensaria em um algoritmo para isso? Vamos ver:

1. Primeiro, terá que pegar um copo;
2. Pegar um recipiente (garrafa, torneira, bacia,...) que contenha a água que quer beber;
3. Transferir a água do recipiente ao copo até uma quantidade suficiente;
4. Por fim, beber a água contida no copo.

Ufa! Cansou só de ler, né? Um algoritmo deve ser algo "mastigado". Se eu falasse "eu vou pegar copo d'agua" o recado estava dado, mas para um computador não é bem assim que funciona. Esses exemplos simples ilustram a ideia de um algoritmo, vamos partir para problemas mais palpáveis.

Lembra das aulas de matemática nas quais você tinha que calcular a raiz de uma equação de primeiro grau? Se você utilizasse algoritmos para resolver este problema sua vida seria bem mais prática, tenho certeza. Vamos ver na próxima seção esse e outros exemplos de como aplicar a representação de algoritmos: para o cálculo de um desconto e para o cálculo da média entre dois valores.

2.2 Representações de Algoritmos

Entre as representações mais comuns de um algoritmo destacaremos nesta seção três: **descrição narrativa**, **fluxograma** e **pseudocódigo**. A principal diferença entre elas está no maior ou menor nível de detalhamento (grau de abstração). Quanto mais detalhes teremos de como implementar esse algoritmo, ou seja, criar um programa a partir da representação mais alto será o grau de abstração. Talvez você possa perguntar: existe uma representação que seja a melhor? A resposta para essa pergunta é: depende!

Cada programador pode optar por um ou outra representação para estruturar o seu algoritmo. Irei definir três problemas para podermos observar cada representação e, assim, conseguirmos analisar e comparar essas representações.

Problema 2.2.1 — Calcular o desconto de um produto. Esse problema é recorrente em muitos casos. Quem já fez aquela pergunta, tem desconto? Levanta a mão! (eu). Vamos criar uma fórmula para isso. Se temos um produto que custa um `valor` em reais e queremos calcular uma `porcentagem` deste `valor` (`desconto`) devemos realizar o seguinte cálculo:

$$\texttt{desconto} = \texttt{valor} \times \frac{\texttt{porcentagem}}{100} \tag{2.1}$$

onde *desconto* é o valor que buscamos.

Por exemplo, qual o desconto de 25% em um produto que custa 150 reais? O resultado seria: $\texttt{desconto} = 150 \times \frac{25}{100}$, logo $\texttt{desconto} = 37{,}50$ reais.

Problema 2.2.2 — Calcular a média entre dois valores. Todo(a) estudante provavelmente deve ter enfrentado este problema, a tal média bimestral. O cálculo da média aritmética entre dois valores consiste em somar os dois valores e dividir por 2 (dois). Formalizando, a `media`, entre dois valores $\texttt{n}_1$ e $\texttt{n}_2$ pode ser calculada com a seguinte fórmula:

$$\texttt{media} = \frac{\texttt{valor}_1 + \texttt{valor}_2}{2} \tag{2.2}$$

Por exemplo, se $\texttt{n}_1 = 8$ e $\texttt{n}_2 = 10$, $\texttt{media} = \frac{8+10}{2}$, fazendo os cálculos a $\texttt{media} = 9$; logo, a média entre esses dois valores, 8 e 10, é 9. Desconsiderem o erro ortográfico de *média* na fórmula, depois saberá o motivo.

Problema 2.2.3 — Calcular a raiz de uma equação de primeiro grau. Toda equação de primeiro grau é uma função que possui uma única incógnita elevada a 1 (usualmente não se coloca o 1). Podemos representar essa função do seguinte modo: $f(\texttt{x}) = \texttt{ax+b}$. Para encontrar a raiz desta função devemos igualá-la a zero e isolar o `x`:

$$\texttt{ax+b} = 0 \tag{2.3}$$

$$\texttt{x} = \frac{-\texttt{b}}{\texttt{a}} \tag{2.4}$$

onde, `x` é o valor que representa a raiz da equação. Observe que o valor de `a` deve diferir de zero ($\texttt{a} \neq 0$), senão teremos uma indeterminação.

Por exemplo, na função $f(\texttt{x}) = 5\texttt{x} - 10$, para encontrar a função faremos $f(\texttt{x}) = 0$, que é a mesma coisa que $5\texttt{x} - 10 = 0$, e isolando `x` temos: $\texttt{x} = 2$. Logo, 2 é a raiz dessa equação.

Com base no entendimento desses problemas vamos agora conhecer cada representação. É imprescindível que você compreenda cada problema. Por isso, leia com calma cada um dos problemas novamente, caso precise, para então prosseguir com as representações.

Este capítulo teve como grande contribuição os trabalhos de (ALGORITMO..., 2004; VASCONCELLOS; TAMARIZ; BATISTA, 2019).

2.2.1 Descrição Narrativa

Na descrição narrativa os algoritmos são expressos diretamente em linguagem natural. Ou seja, uma sequência de passos é descrita em nossa língua. É só lembrar dos nossos exemplos na Seção 2.1, rotina e beber água, onde utilizamos esse tipo de descrição.

Esta representação não é tão boa por permitir várias interpretações, dificultando transcrição para um programa. Por exemplo, se eu falar "rebolar no mato" um piauiense, ou cearense, possivelmente saberia o significado: *jogar no lixo*. Porém, pode acontecer de alguém de fora destas regiões possa não compreender bem e pensar que alguém pessoa irá

dançar, rebolar, em uma área com mato.

Agora vamos aos exemplos ordenados conforme os Problemas 2.2.1, 2.2.2 e 2.2.3.

■ **Exemplo 2.1** Problema 2.2.1 (Desconto):

1. Obter `valor`.
2. Obter `porcentagem` (de 0 a 100).
3. Calcular o `desconto`.

■

■ **Exemplo 2.2** Problema 2.2.2 (Média Aritmética):

1. Obter o valor da primeira nota, $\mathtt{n}_1$.
2. Obter o valor da segunda nota, $\mathtt{n}_2$.
3. Calcular a média (`media`).

■

■ **Exemplo 2.3** Problema 2.2.3 (Raiz da Equação de Primeiro Grau):

1. Obter o valor de `a`.
2. Obter o valor de `b`.
3. Se `a` é igual a zero, dizer que não existe raiz para a equação.
4. Senão, a raiz da equação será $-\frac{\mathtt{b}}{\mathtt{a}}$.

■

Bem simples, né!? Observa-se que com a descrição narrativa podemos facilmente resolver nossos problemas. Porém, em algumas situações não foram detalhados alguns procedimentos, como no cálculo do desconto (Exemplo 2.1).

2.2.2 Fluxograma

É uma representação gráfica em que cada forma descreve uma ação distinta (instruções, comandos, direções, etc). Por se tratar de um sistema de representação visual simplificado é mais precisa que a descrição narrativa, pois minimiza a ambiguidade. É como uma placa de trânsito, você sabe seu significado de imediato ao vê-la (embora muitos confundam as placas de proibido estacionar e de proibido parar e estacionar). Aqui, começaremos a ver a silhueta da estrutura de um programa, com *comandos* de entrada, saída, e atribuição de dados, além de comandos de decisão (condicionais) (Figura 2.1). Além disso, temos um símbolo que representará a direção do fluxo, uma seta.

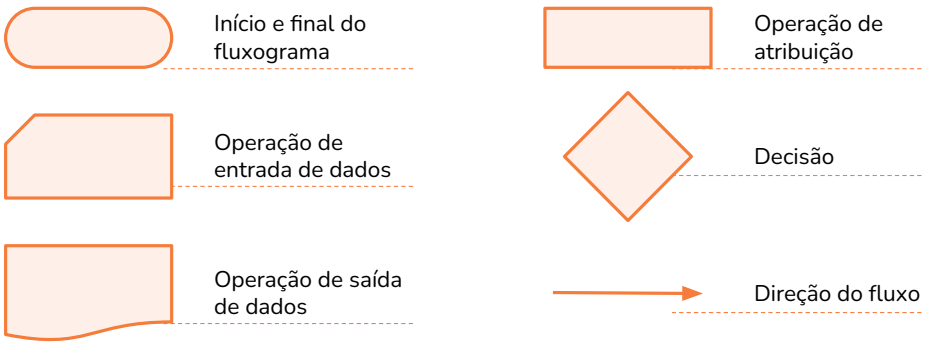


Figura 2.1: Símbolos presentes na representação de fluxograma e seus significados.

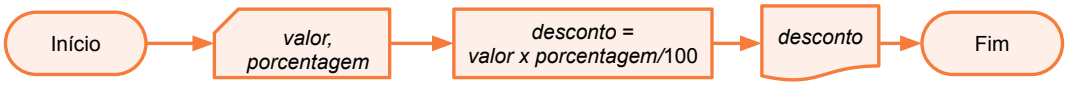
Os símbolos que usaremos estão dispostos na Figura 2.1. Cada símbolo tem seu significado próprio. Em um fluxograma sempre existirá um início e pelo menos um fim. Posso destacar que as operações que serão recorrentemente utilizadas são: operação de

entrada de dados, onde o usuário irá informar um valor e o mesmo será atribuído a uma variável; operação de saída de dados onde algo será impresso (exibido na tela); e operação de atribuição, onde um valor será atribuído a uma variável.

O símbolo de decisão, losango, indica que o fluxo será dividido em duas partes, caso seja verdadeiro seguirá uma direção, caso contrário seguirá outra direção.

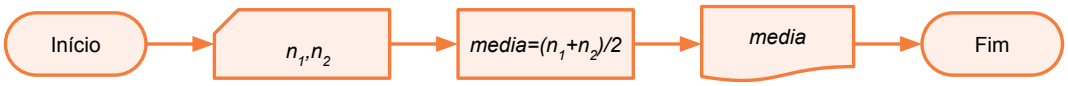
Vejamos cada um dos problemas agora utilizando fluxogramas.

■ **Exemplo 2.4** Problema 2.2.1 (Desconto):



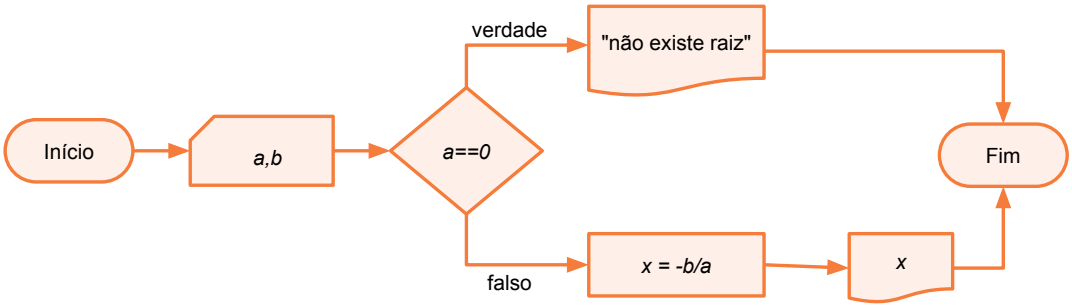
Observe que podemos obter mais de um valor utilizando uma vírgula ao invés de usar duas formas para entrada de dados. Após o cálculo do desconto ele será impresso pelo algoritmo, caso não tenha uma operação de impressão nunca saberemos o que foi realizado pelo algoritmo. ■

■ **Exemplo 2.5** Problema 2.2.2 (Média Aritmética):



■

■ **Exemplo 2.6** Problema 2.2.3 (Raiz da Equação de Primeiro Grau):



Aqui temos um símbolo de decisão. A representação de igualdade é feita com dois símbolos de igual juntos (==). Caso seja verdade, ou seja, `a` é igual a zero, a raiz não existe e seguimos a direção da impressão do texto, delimitado com aspas duplas, informando isso. ■

Observa-se que o fluxo descreve o passo a passo do algoritmo. Essa forma de representação é excelente para visualizar como o algoritmo deve funcionar.

2.2.3 Pseudocódigo

Esta representação é rica em detalhes, tanto na estrutura como na definição dos tipos das variáveis usadas no algoritmo. Aqui, utiliza-se também a linguagem natural para estruturar a representação em conjunto com palavras-chave que serão os *comandos*.

Entre os comandos destaco os seguintes:

- **Entrada**: definição das variáveis de entrada;
- **Saída**: definição da saída do algoritmo;
- **leia**: indica uma operação de entrada de dados;
- **imprime**: indica uma operação de saída de dados;
- $\leftarrow$: significa uma atribuição;
- **se**: verifica se uma condição é *verdadeira*, caso seja é executado o que tiver dentro deste escopo;
- **senão**: define um escopo que será executado caso a condição do **se** tiver sido *falsa*. Um **senão** só pode ser definido associado a um **se**.

■ **Exemplo 2.7** Problema 2.2.1 (Porcentagem): ■

Algoritmo 2.1: Calcula Desconto

Entrada: *valor*, *porcentagem*
Saída: *desconto*
1 $desconto \leftarrow valor \times \frac{porcentagem}{100}$
2 **retorna** *desconto*

No Exemplo 2.7, defini na **Entrada** quais os valores que precisamos possuir para que o algoritmo funcione. E em **Saída**, defini o que o algoritmo irá retornar, para ser impresso na tela, por exemplo. Esses comandos poderiam ser substituídos por **leia** e **imprima**, respectivamente.

■ **Exemplo 2.8** Problema 2.2.2 (Média Aritmética): ■

Algoritmo 2.2: Calcula Média de Dois Valores

1 **leia** $nota1$
2 **leia** $nota2$
3 $media \leftarrow \frac{nota1 + nota2}{2}$
4 **imprime** *media*

No Exemplo 2.8, o nome da variável sem o acento (*media*) não foi um descuido. Vamos nos acostumar a não acentuar as nossas variáveis, pois as linguagens de programação não entendem alguns símbolos, veremos isso com detalhes no próximo capítulo.

■ **Exemplo 2.9** Problema 2.2.3 (Raiz da Equação de Primeiro Grau):

Algoritmo 2.3: Calcula Raiz da Equação de Primeiro Grau

Entrada: a, b
Saída: x
1 **se** $(a == 0)$ **então**
2 | **imprime** "não existe raiz"
3 | **retorna**
4 **senão**
5 | $x \leftarrow -\frac{b}{a}$
6 | **retorna** x

■

No Exemplo 2.9, o símbolo de == é um operador lógico de igualdade, ele indica se a igualdade é verdadeira ou falsa. Por exemplo, $5 == 2$ retorna *falso*, e $7 == 7$ retorna *verdadeiro*. No caso, na linha 1 está sendo verificado se a é igual a 0. Se for igual a zero será executado o escopo do **se** (linhas 2 e 3), pois a condição é *verdadeira*; caso contrário será executado o escopo do **senão** (linhas 5 e 6). Podemos encarar a condição $(a == 0)$ como uma premissa (algo verdadeiro) para que o escopo do **se** seja executado.

Oriento que tente realizar mais o uso do pseudocódigo para se familiarizar com a estrutura dos programas que faremos utilizando a linguagem de programação *javascript*.

2.3 Resumo

Podemos agora observar a Tabela 2.1 com olhar mais criterioso enxergando as vantagens e desvantagens de cada abordagem. Em geral, os programadores utilizam mais as representações de fluxograma e pseudocódigo para construção de seus algoritmos por direcionar melhor a construção do programa, principalmente o pseudocódigo.

Representação	Grau de Abstração	Vantagens	Desvantagens
Descrição Narrativa	Baixo	A língua natural é conhecida, não é preciso entender novas palavras e linguagens	A língua natural por se só abre margem para ambiguidades, dificultando a construção do programa a partir do algoritmo
Fluxograma	Médio	Conhecendo os símbolos (gráficos) é mais fácil entender o fluxo do algoritmo	Sendo uma versão simplificada do algoritmo o fluxo pode não possuir detalhes suficientes para construção do programa
Pseudocódigo	Alto	Representação clara mesmo sem conhecer as especificações de linguagem de programação	As regras do pseudocódigo devem ser conhecidas

Tabela 2.1: Resumo das representações de um algoritmo com vantagens e desvantagens.

2.4 Exercícios

Exercício 2.1 Entre os exemplos abaixo, NÃO pode ser considerado um algoritmo:

a) Guia de montagem de um guarda-roupas.
b) Manual de instruções de uso do celular.
c) Receita de sorvete.
d) Cardápio da pizzaria.

■

Exercício 2.2 A afirmação "O algoritmo é uma sequência de passos lógicos e finitos e não-ambíguos que permitem solucionar problemas" é:

a) Verdadeira.

b) Falsa.

■

Exercício 2.3 A afirmação "É um consenso entre os programadores que a melhor forma de representação de um algoritmo é a descrição narrativa" é:

a) Verdadeira.
b) Falsa.

■

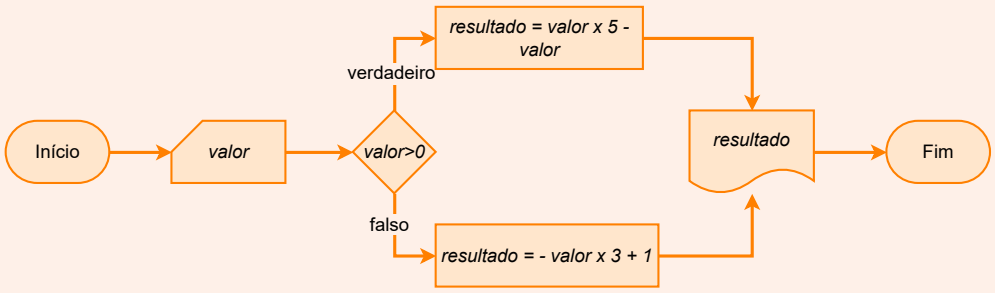
Exercício 2.4 Faça o algoritmo que verifique se uma pessoa é maior de idade conforme a idade informada. Utilize qualquer uma das representações para desenvolver seu algoritmo. ■

Exercício 2.5 Faça um algoritmo que verifica se um número informado é positivo ou negativo. Utilize qualquer uma das representações para desenvolver seu algoritmo. ■

Exercício 2.6 Faça um algoritmo que verifica se a média aritmética de três notas aprova ou reprova um(a) aluno(a). O(A) aluno(a) é considerado(a) aprovado(a) se sua média for maior ou igual a 7. Utilize qualquer uma das representações para desenvolver seu algoritmo. ■

Exercício 2.7 O ***jokenpo*** é um jogo que é jogado entre duas pessoas colocando uma configuração de mão: Pedra (✊), Papel (✋) e Tesoura (✌). As regras são as seguintes: ✌ ganha de ✋ e perde para ✊; ✊ ganha de ✌ e perde para ✋; e ✋ ganha de ✊ e perde para ✌. Caso os jogadores coloquem a mesma configuração de mão é empate. Caso um dos jogadores ganhe você deve informar qual jogador ganhou. Utilize qualquer uma das representações para desenvolver seu algoritmo. ■

Exercício 2.8 Utilizando o fluxograma a seguir responda cada um dos itens.



a) Se o *valor* de entrada for 5, o que irá ser impresso?
b) Se o *valor* de entrada for -2, o que irá ser impresso?
c) Se o *valor* de entrada for 0 (zero), o que irá ser impresso?
d) Faça um pseudocódigo a partir deste fluxograma.

■