

Técnicas de Programação Algorítmica

Organizador Sérgio Guedes

Páginas de 3 a 12

ele, inserindo informações e visualizando resultados. Tudo isso reunido possibilita que ele realize sua principal tarefa: processar dados.

Mas como é que acontece esse processamento de dados? Bem, para realizar esse processamento, ele precisa receber dados fornecidos por um dispositivo de entrada, como um teclado, mouse, microfone ou scanner, para então realizar um conjunto de operações com ou sobre esses dados e, assim, gerar uma resposta por meio de dispositivos de saída, como uma impressora, um monitor ou caixas de som.

No entanto, mesmo sendo poderoso e magnífico, como já dissemos, nada disso garante que ele desempenhe suas tarefas e que o resultado seja efetivamente útil!

Aqui entramos nós, não tão poderosos e magníficos, para fornecer ao computador o que ele precisa ter para efetivamente produzir algo de real utilidade. Precisamos colocar no computador um conjunto de especificações claras e detalhadas de como e em quais situações ele deve agir sobre um conjunto de dados. Ou seja, precisamos programá-lo para exercer suas tarefas e solucionar um problema ou, do contrário, ele não saberá agir.

Sabendo que precisamos ser claros, diretos, objetivos e imperativos, necessitamos efetuar uma programação com uma sequência precisa e bem definida de instruções, de tal modo que cada uma delas possa ser executada em um período finito e use adequadamente a capacidade de processamento do computador. A essa elaboração de sequência precisa e bem definida damos o nome de algoritmo!

Apresentação e introdução a algoritmos

É provável que você ainda não tenha clareza acerca do conceito de algoritmo. Não há nenhum problema nisso. Mas reflita sobre esta afirmação: certamente você cria e executa diversos deles diariamente, sem nem ao menos se dar conta disso. Neste tema explicaremos, de forma simples, o que são algoritmos, seu conceito, diferentes tipos e a aplicação de cada um deles na computação e também na vida cotidiana.

O que é algoritmo?

Apesar do nome incomum, *algoritmo* nada mais é do que uma sequência de passos que devem ser realizados para alcançar determinado objetivo.

Podemos encontrar diversos exemplos de algoritmo em nosso dia a dia, como aquela receita de bolo que te dá água na boca (e na nossa também!). Nela está descrito um conjunto de ingredientes necessários, com suas respectivas quantidades e uma sequência de passos (ações) que devem ser cumpridos fielmente (a menos que você seja um *chef* de cozinha) para que o resultado (objetivo) final seja atingido com sucesso.



Saiba mais

Quando criamos um algoritmo, devemos especificar ações claras e precisas que, a partir de um estado inicial e após certo tempo, produzam um resultado final previsível. Ou seja, um algoritmo estabelece um padrão de comportamento a ser seguido, com o objetivo de solucionar algum problema garantindo que, sempre que for feito da mesma forma, o resultado final será o mesmo.

Independentemente da linguagem utilizada, o algoritmo é um conceito básico, encontrado também em nosso cotidiano. Cada tarefa que realizamos utiliza um algoritmo, que funciona como uma receita, seguindo os passos necessários para chegar a determinado resultado final (solução de um problema) que, a princípio, deverá ser sempre o mesmo. Por exemplo, vamos pensar em uma tarefa cotidiana como fazer uma limonada. Para isso, devemos passar por algumas etapas:

1. escolher os limões;
2. cortar os limões;
3. espremer os limões;
4. adicionar água gelada ao suco do limão;
5. adoçar.

Perceba que, ao seguir esses passos, obteremos um resultado final que, no caso, é a limonada. Se seguirmos sempre os mesmos passos, o resultado final será invariavelmente a limonada. Isso que acabamos de fazer é um algoritmo, que se aplica a uma tarefa cotidiana qualquer. Ao longo deste livro, estudaremos, de forma clara e didática, os tipos e conceitos de algoritmo e programação, suas aplicações, funcionalidades e métodos de construção. Faremos sempre uma comparação clara com o nosso dia a dia, assim

como fizemos anteriormente com o exemplo da limonada, tornando o aprendizado mais simples e ilustrado. Para melhor ilustrar o que são algoritmos, veremos a seguir alguns exemplos.



Exemplo

1. Somar três números:
 - Passo 1 – receber os três números.
 - Passo 2 – somar os três números.
 - Passo 3 – mostrar o resultado obtido.
2. Fazer um sanduíche:
 - Passo 1 – pegar o pão.
 - Passo 2 – cortar o pão.
 - Passo 3 – pegar a maionese.
 - Passo 4 – passar a maionese no pão.
 - Passo 5 – pegar e cortar alface e tomate.
 - Passo 6 – colocar alface e tomate no pão.
 - Passo 7 – pegar o hambúrguer.
 - Passo 8 – fritar o hambúrguer.
 - Passo 9 – colocar o hambúrguer no pão.
3. Trocar uma lâmpada:
 - Passo 1 – pegar uma lâmpada nova.
 - Passo 2 – testar a lâmpada.
 - Passo 3 – pegar uma escada.
 - Passo 4 – posicionar a escada embaixo da lâmpada queimada.
 - Passo 5 – subir na escada com a lâmpada nova na mão.
 - Passo 6 – retirar a lâmpada queimada.
 - Passo 7 – colocar a lâmpada nova.
 - Passo 8 – descer da escada.
 - Passo 9 – testar o interruptor.
 - Passo 10 – guardar a escada.
 - Passo 11 – embrulhar a lâmpada velha com jornal.
 - Passo 12 – jogar a lâmpada velha no lixo de vidros.
4. Ir para a escola:
 - Passo 1 – acordar cedo.
 - Passo 2 – reclamar que é muito cedo.
 - Passo 3 – ir ao banheiro.

- Passo 4 – abrir o armário para escolher uma roupa.
 - Passo 5 – se o tempo estiver quente, pegar uma camiseta e uma calça jeans; caso contrário, pegar um agasalho e uma calça jeans.
 - Passo 6 – vestir a roupa escolhida.
 - Passo 7 – tomar café.
 - Passo 8 – sair de casa.
 - Passo 9 – se a escola não for perto, pegar uma condução ou a bicicleta e descer perto da escola; senão, ir a pé.
 - Passo 10 – entrar na escola.
5. Sacar dinheiro no banco 24 horas:
- Passo 1 – ir até um caixa 24 horas.
 - Passo 2 – inserir o cartão.
 - Passo 3 – digitar a senha.
 - Passo 4 – verificar o saldo.
 - Passo 5 – se o saldo for maior ou igual à quantia desejada, fazer o saque; senão, sair do banco 24h (triste).

É natural que você pense: “Mas eu realizo essas atividades de maneira diferente!”.

É claro que sim, mas isso ocorre justamente porque os problemas podem ser resolvidos de muitas formas, obtendo-se a mesma resposta. Ou seja, podem existir vários algoritmos para solucionar um único problema.

Construindo algoritmos

Para construir um algoritmo, é preciso seguir alguns passos:

1. Entender o problema a ser resolvido e destacar os pontos mais importantes e os objetos que o compõem.
2. Definir os dados de entrada, ou seja, quais dados serão fornecidos e quais objetos fazem parte do cenário do problema.
3. Definir o processamento, ou seja, quais operações serão efetuadas e quais as restrições para essas operações. O processamento deve transformar os dados de entrada em dados de saída e também verificar quais objetos são responsáveis pelas atividades.
4. Definir os dados de saída, ou seja, quais dados serão gerados depois do processo.
5. Construir o algoritmo utilizando um dos tipos apresentados a seguir no tópico “Tipos de algoritmo”.

6. Testar o algoritmo realizando simulações.
7. Corrigir possíveis erros e voltar ao item 5.

Tipos de algoritmo

Existem três tipos mais comuns de algoritmo: descrição narrativa, fluxograma e pseudocódigo/portugol.

Descrição narrativa

Consiste em analisar o problema e escrever, utilizando uma linguagem natural, os passos para sua resolução.





- **Vantagem** – não é necessário conhecer nenhum conceito novo.
- **Desvantagem** – a língua natural pode ser interpretada de várias formas, o que dificultará a transcrição para um programa.


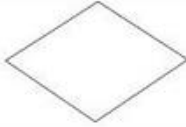


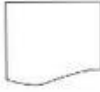
Fluxograma

Consiste em analisar o problema e escrever, utilizando símbolos gráficos predefinidos (Tabela 1.1), os passos para sua resolução.

- **Vantagem** – símbolos gráficos são mais simples de compreender do que textos.
- **Desvantagem** – é preciso aprender a simbologia e não fornece detalhes mais precisos, dificultando uma transcrição para uma linguagem de programação. Além do mais, problemas complexos resultam em um desenho gráfico muito denso que torna difícil a visualização.

Tabela 1.1 Conjunto de símbolos utilizados em fluxogramas.

Símbolos	Função
	Indica o início ou fim de um fluxograma (algoritmo).
	Indica o sentido do fluxo de dados, conectando os símbolos ou blocos existentes.
	Indica operações matemáticas e de manipulação de textos, palavras, letras.
	Representa entrada de dados (que não sejam digitados).

Símbolos	Função
	Representa entrada manual de dados (digitação).
	Indica uma tomada de decisão com duas opções lógicas.
	Indica uma mensagem em dispositivo de vídeo.
	Indica uma preparação, um ciclo de repetição.
	Indica uma impressão de relatório.

Pseudocódigo ou português

Consiste em analisar o problema e escrever, por meio de regras, os passos para sua resolução.

- **Vantagem** – a passagem para qualquer linguagem de programação é quase imediata.
- **Desvantagem** – a princípio, podemos dizer que não há desvantagens, mas elas existem! Primeiro é necessário lidar com a lógica de programação e, então, aprender as regras do padrão de pseudocódigo utilizado. Ou seja, na prática, ao criar um pseudocódigo, você estará desenvolvendo o programa aplicativo para depois transcrevê-lo para uma linguagem de programação.

Apesar dessa desvantagem, o pseudocódigo é amplamente utilizado por todos os programadores, analistas de sistemas, matemáticos, físicos e outros, para descrever seus algoritmos. No caso da computação, é mais comum utilizar um pseudocódigo, por ser mais próximo de uma linguagem do dia a dia, embora bastante simplificada.

Exemplos de algoritmo

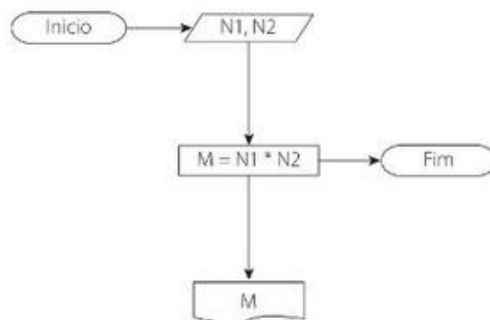
Vejamos exemplos de implementação de algoritmos em cada tipo citado.

Multiplicar dois números

Descrição narrativa

- Passo 1 – receber dois números que serão multiplicados.
- Passo 2 – multiplicar.
- Passo 3 – mostrar o resultado.

Fluxograma 1.1



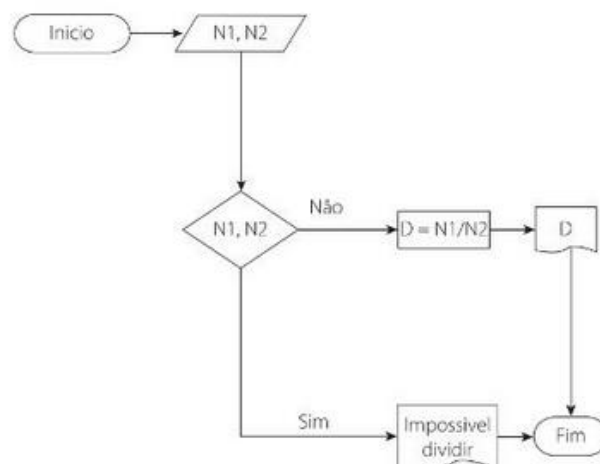
Pseudocódigo 1.1

```
Var
    N1, N2, M: Numerico
Inicio
    Escreva "Digite dois números"
    Leia (N1, N2)
    M ← (N1 * N2)
    Escreva ("Multiplicação =", M)
Fimalgoritmo
```

Mostrar a divisão de dois números

Descrição narrativa

- Passo 1 – receber os dois números que serão divididos.
- Passo 2 – se o segundo número for igual a zero, não poderá ser feita a divisão, pois não existe divisão por zero; caso contrário, dividir os números e mostrar o resultado.

Fluxograma 1.2**Pseudocódigo 1.2**

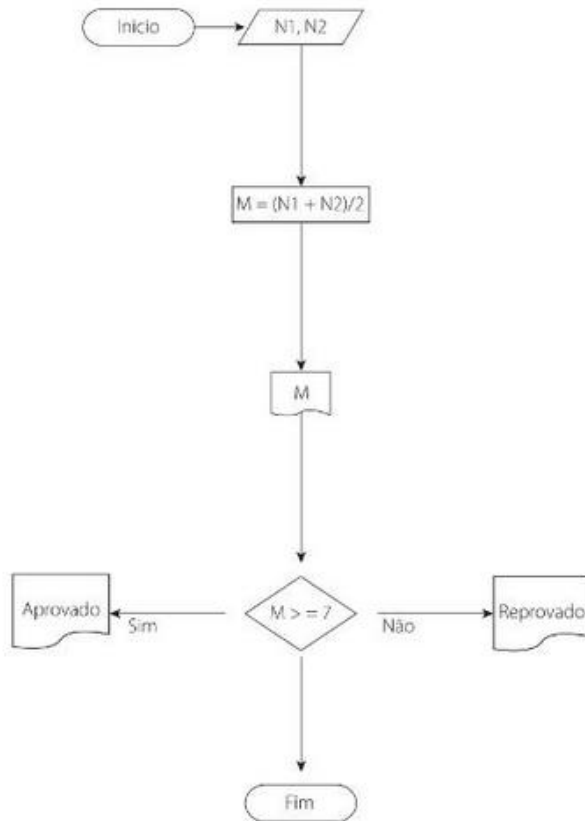
```

Var
    N1, N2, D: Inteiro
Início
    Escreva ("Digite dois números")
    Leia (N1, N2)
    Se (N2 = 0) Então
        Escreva ("Impossível dividir")
    Senão
        D ← N1/N2;
        Escreva ("Divisão = ", D)
    Fimse
Fimalgoritmo
  
```

Calcular a média aritmética entre duas notas de um aluno e mostrar sua situação, que pode ser aprovado ou reprovado

Descrição narrativa

- Passo 1 – receber as duas notas.
- Passo 2 – calcular a média aritmética.
- Passo 3 – mostrar a média aritmética.
- Passo 4 – se a média for maior ou igual a 7, então o aluno estará aprovado; caso contrário, reprovado.

Fluxograma 1.3**Pseudocódigo 1.3**

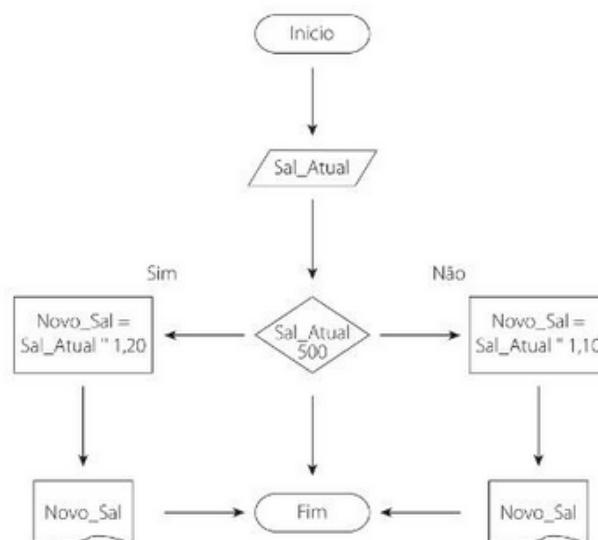
```
Var
    N1, N2, M: inteiro
Início
    Escreva ("Digite as duas notas")
    Leia (N1, N2)
    M ← (N1 + N2) / 2
    Escreva ("Média = ", M)
    Se (M >= 7) Então
        Escreva ("Aprovado")
    Senão
        Escreva ("Reprovado")
    Fimse
Fimalgoritmo
```

Calcular o novo salário de um funcionário, sabendo que quem recebe salário de até R\$ 500,00 terá aumento de 20%, os outros terão de 10%

Descrição narrativa

- Passo 1 – receber o salário atual do funcionário.
- Passo 2 – se o salário for de até R\$ 500,00, calcular o novo salário com aumento de 20%; caso contrário, calcular com aumento de 10%.

Fluxograma 1.4



Pseudocódigo 1.4

```
Var
    Sal_Atual, Novo_Sal: inteiro;
Início
    Escreva "Digite o salário atual do funcionário"
    Leia Sal_Atual
    Se Sal_Atual <= 500 Então
        Novo_Sal ← Sal_Atual * 1,20
    Senão
        Novo_Sal ← Sal_Atual * 1,10
    Fimse
    Escreva ("Novo salário = ", Novo_Sal)
Fimalgoritmo
```