algoritmos e programação

com exemplos em Pascal e C

nina edelweiss
maria aparecida castro livi

bookman

as autoras

Nina Edelweiss é engenheira eletricista e doutora em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. Durante muitos anos, lecionou em cursos de Engenharia e de Ciência da Computação na UFRGS, na UFSC e na PUCRS. Foi, ainda, orientadora do Programa de Pós-Graduação em Ciência da Computação da UFRGS. É coautora de três livros, tendo publicado diversos artigos em periódicos e em anais de congressos nacionais e internacionais. Participou de diversos projetos de pesquisa financiados por agências de fomento como CNPq e FAPERGS, desenvolvendo pesquisas nas áreas de bancos de dados e desenvolvimento de software.

Maria Aparecida Castro Livi é licenciada e bacharel em Letras, e mestre em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. Desenvolveu sua carreira profissional na UFRGS, onde foi programadora e analista de sistema, antes de ingressar na carreira docente. Ministrou por vários anos a disciplina de Algoritmos e Programação para alunos dos cursos de Engenharia da Computação e Ciência da Computação. Sua área de interesse prioritário é o ensino de Linguagens de Programação, tanto de forma presencial quanto a distância.

E22a Edelweiss, Nina.
Algoritmos e programação com exemplos em Pascal e C [recurso eletrônico] / Nina Edelweiss, Maria Aparecida Castro Livi. – Dados eletrônicos. – Porto Alegre : Bookman, 2014.

Editado também como livro impresso em 2014.
ISBN 978-85-8260-190-7

1. Informática. 2. Algoritmos – Programação. I. Livi, Maria Aparecida Castro. II. Título.

CDU 004.421

Catalogação na publicação: Ana Paula M. Magnus – CRB 10/2052

4.3 → comando de seleção dupla

Supondo que, além de informar a média das três notas do aluno, também se queira que o programa informe se ele foi aprovado (quando a média for igual ou superior a 6) ou reprovado (média inferior a 6). Dois comandos, `se-então`, são necessários para imprimir essas mensagens:

```
se média ≥ 6
então escrever('Aprovado')      {INFORMA SE O ALUNO FOI APROVADO}
se média < 6
então escrever('Reprovado')     {INFORMA SE O ALUNO FOI REPROVADO}
```

Os dois comandos implementam ações mutuamente exclusivas e dependem da avaliação de uma mesma condição, sendo uma das ações associada ao resultado verdadeiro e outra ao resultado falso. Para evitar a repetição da comparação, pode ser utilizado um **comando de seleção dupla** que, a partir do resultado da avaliação de uma condição, seleciona um de dois comandos para ser executado. Sua sintaxe é:

```
se <expressão lógica>
então <comando>
senão <comando>
```

Somente um comando pode ser definido em cada uma das cláusulas `então` e `senão`. Esse comando pode ser simples ou composto, como no caso do comando de seleção simples. O exemplo anterior, resolvido através de dois comandos de seleção simples, equivale ao seguinte comando de seleção dupla:

```
se média ≥ 6
então escrever('Aprovado')
senão escrever('Reprovado')
```

O fluxograma que representa esse comando, mostrado na Figura 4.4, mostra claramente que o fluxo do programa passa por apenas um dos dois comandos, o qual é selecionado pelo resultado da expressão lógica.

O algoritmo que calcula a média de três notas e informa se o aluno foi aprovado ou reprovado é o seguinte:

```
Algoritmo 4.3 - Média4
{INFORMA A MÉDIA DO ALUNO E SE FOI APROVADO OU REPROVADO}
  Entradas: nota1, nota2, nota3 (real)
  Saídas: média (real)
          {Informação de aprovado ou reprovado}
início
  ler (nota1, nota2, nota3)     {ENTRADA DAS 3 NOTAS}
  média ← (nota1+nota2+nota3)/3
  escrever (média)              {INFORMA MÉDIA CALCULADA}
```

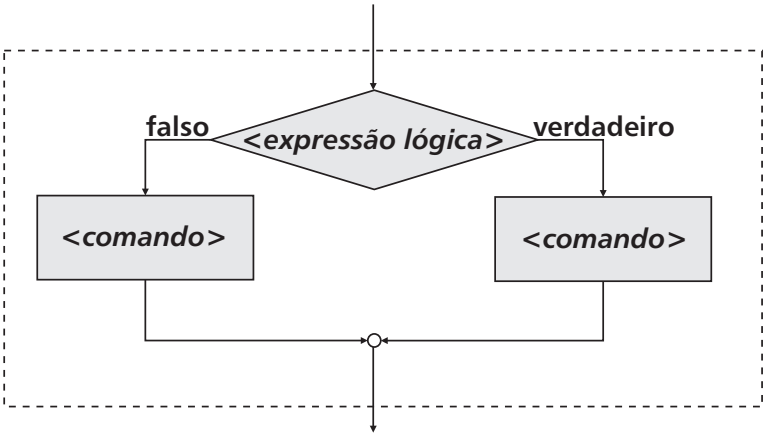

figura 4.4 Fluxograma do comando de seleção dupla.

```
  se média ≥ 6
  então escrever('Aprovado')    {INFORMA SE O ALUNO FOI APROVADO}
  senão escrever('Reprovado')   {INFORMA SE O ALUNO FOI REPROVADO}
fim
```

4.4 → comandos de seleção aninhados

Conforme visto, somente um comando pode ser utilizado nas cláusulas `então` e `senão`, mas não há restrição quanto ao tipo de comando. Pode, inclusive, ser usado um novo comando de seleção – simples ou dupla. Nesse caso, diz-se que os comandos de seleção estão aninhados ou encadeados.

Estendendo um pouco mais a aplicação de apoio a um professor, suponha que se queira obter o conceito do aluno com base na sua média, de acordo com a seguinte conversão:

Conceito A: Média ≥ 9,0
Conceito B: 9,0 > Média ≥ 7,5
Conceito C: 7,5 > Média ≥ 6,0
Conceito D: Média < 6,0

A solução pode ser implementada através de uma sequência de comandos condicionais (opção 1):

```
se média ≥ 9
então conceito ← 'A'
se (média < 9) e (média ≥ 7,5)
então conceito ← 'B'
se (média < 7,5) e (média ≥ 6,0)
então conceito ← 'C'
se (média < 6)
então conceito ← 'D'
```

Nessa sequência de comandos, somente uma das condições será verdadeira e, apesar disso, todas as condições serão sempre avaliadas, desnecessariamente. Para evitar isso, o algoritmo a seguir calcula a média e o conceito, utilizando comandos de seleção dupla aninhados (opção 2). Note que, uma vez encontrada uma condição verdadeira, as que estão após ela, na cláusula senão, não são mais avaliadas. Cabe ressaltar que, nessa solução, não foi feita a análise da validade dos dados de entrada, partindo-se do pressuposto que eles foram corretamente informados.

```
Algoritmo 4.4 - MédiaConceito1
{INFORMA A MÉDIA E O CONCEITO DE UM ALUNO}
  Entradas: nota1, nota2, nota3 (real)
  Saídas: média (real)
          conceito (caractere)
início
  ler (nota1, nota2, nota3)       {ENTRADA DAS 3 NOTAS}
  média ← (nota1+nota2+nota3)/3   {CÁLCULO DA MÉDIA}
  escrever (média)                {INFORMA MÉDIA CALCULADA}
  se média ≥ 9                    {CÁLCULO DO CONCEITO}
  então conceito ← 'A'
  senão se média ≥ 7,5
        então conceito ← 'B'
        senão se média ≥ 6,0
              então conceito ← 'C'
              senão conceito ← 'D'  {MÉDIA < 6}
  escrever (conceito)             {INFORMA CONCEITO}
fim
```

Nesse algoritmo, o trecho de programa que calcula o conceito corresponde a um único comando de seleção dupla. Se a média for igual ou superior a 9,0, o conceito "A" é atribuído ao aluno e a execução desse comando termina. No caso dessa condição não ser verdadeira, então é avaliada a segunda condição, que verifica se a média é igual ou superior a 7,5. Se essa condição for verdadeira, o aluno recebe o conceito "B" e o comando é concluído. Se não for verdadeira, então a média é novamente analisada, dessa vez verificando se é maior ou igual a 6,0. Finalmente, independentemente da condição ser verdadeira ou falsa, o comando é encerrado com a atribuição do conceito "C" (expressão verdadeira) ou "D" (expressão falsa).

A compreensão do funcionamento dos comandos de seleção aninhados é bem mais clara do que a da sequência de comandos condicionais (opção 1). Além disso, a segunda opção de representação realiza menos comparações do que a primeira, o que diminui o tempo de execução.

uso de indentação para delimitar comandos aninhados. A pseudolinguagem utilizada neste livro faz uso da indentação para mostrar visualmente o escopo de cada um dos coman-

dos de seleção aninhados. Sem a indentação, é bem mais difícil visualizar o funcionamento dos comandos aninhados, como pode ser observado na reescrita do trecho do Algoritmo 4.4 que examina a média:

```
se média ≥ 9
então conceito ← 'A'
senão se média ≥ 7,5
então conceito ← 'B'
senão se média ≥ 6,0
então conceito ← 'C'
senão conceito ← 'D'  {MÉDIA < 6}
```

Contudo, a indentação por si só não garante a correção do código e pode até mesmo mascarar erros se não corresponder à sintaxe do código utilizado. No trecho a seguir, no comando de seleção dupla, o comando da cláusula `então` é um comando condicional. A indentação utilizada faz crer que a cláusula `senão` pertence ao comando mais externo, quando, pela sintaxe, ela pertence ao mais interno:

```
se nota1 = 10      {COMANDO DE SELEÇÃO DUPLA}
então se média > 9 {COMANDO CONDICIONAL}
      então escrever ('Parabéns pela boa média!')
senão escrever ('A primeira nota não é 10! ')
```

A indentação que reflete a sintaxe do que está escrito é:

```
se nota1 = 10      {COMANDO SELEÇÃO DUPLA TRATADO COMO CONDICIONAL}
então se média > 9 {COMANDO CONDICIONAL TRATADO COMO SELEÇÃO DUPLA}
      então escrever ('Parabéns pela boa média!')
      senão escrever ('A primeira nota não é 10!')
```

Da forma como está o trecho, independentemente da indentação utilizada, se a `Nota1` fornecida for 10 e a média não for superior a 9, será produzida a mensagem 'A primeira nota não é 10!', que claramente está incorreta. Nesse caso, o problema pode ser corrigido através do uso dos delimitadores de um comando composto, para indicar que somente a cláusula `então` faz parte do comando que testa a condição "`se média > 9`":

```
se nota1 = 10      {COMANDO DE SELEÇÃO DUPLA}
então início       {COMANDO COMPOSTO}
      se média > 9 {COMANDO CONDICIONAL}
      então escrever ('Parabéns pela boa média! )
      fim
senão escrever ('A primeira nota não é 10!')
```

exercício 4.2 Dados os coeficientes de uma equação do 2º grau, calcular e informar os valores de suas raízes.

```
Algoritmo EquaçãoSegundoGrau
{INFORMA OS VALORES DAS RAÍZES DE UMA EQUAÇÃO DO SEGUNDO GRAU}
  Entradas: a, b, c (real)              {COEFICIENTES DA EQUAÇÃO}
  Saídas: r1, r2 (real)                 {RAÍZES}
  Variável auxiliar: disc (real)        {DISCRIMINANTE}
início
  ler(a, b, c)  {ENTRADA DOS VALORES DOS COEFICIENTES DA EQUAÇÃO}
  se a = 0
  então início
        escrever('Não é equação do segundo grau! ')
        escrever('Raiz = ', (- c / b ))
        fim
  senão início
        disc ← sqr(b) - 4 * a * c       {CÁLCULO DO DISCRIMINANTE}
        se disc < 0
        então escrever('Raízes imaginárias !')
        senão início
              r1 ← ( - b + sqrt ( disc ) ) / ( 2 * a )
              r2 ← ( - b - sqrt ( disc ) ) / ( 2 * a )
              escrever('Raízes: ', r1, r2)
              fim
       fim
fim
```

exercício 4.3 Processar uma venda de livros em uma livraria. O cliente poderá comprar diversas unidades de um mesmo tipo de livro. O código que define o tipo do livro vendido (A, B, C) e o número de unidades desse livro são fornecidos como dados de entrada.

Preços: Tipo A – R$ 10,00
Tipo B – R$ 20,00
Tipo C – R$ 30,00

Calcular e informar o preço a pagar. Caso tenham sido vendidos mais de 10 livros, exibir uma mensagem informando isso.

A solução deste problema é mostrada em duas etapas. Inicialmente, é apresentado um algoritmo em passos gerais para dar uma visão global da solução. Depois, cada um dos passos é detalhado, dando origem ao algoritmo completo.

```
Algoritmo UmaVenda - PASSOS GERAIS
{PROCESSA UMA VENDA DE LIVROS}
  Entradas: tipo do livro ('A', 'B' ou 'C')
            número de livros
  Saídas: preço a pagar
          mensagem indicando que foram vendidas mais de 10 unidades
```

1. Obter dados
2. Calcular preço de venda
3. Emitir mensagem caso necessário
4. Terminar

Detalhamento do algoritmo:

```
Algoritmo UmaVenda
{PROCESSA UMA VENDA DE LIVROS}
  Entradas: código (caractere)          {CÓDIGO DO LIVRO}
            numeroUnidades (inteiro)    {NR. UNIDADES VENDIDAS}
  Saídas: aPagar (real)                 {PREÇO A PAGAR}
          {Mensagem indicando que foram vendidas mais de 10
           unidades}
início
  ler(código, numeroUnidades)           {ENTRADA DE DADOS}
  se código = 'A'                       {CÁLCULO DO PREÇO DA VENDA}
  então aPagar ← numeroUnidades * 10
  senão se código = 'B'
        então aPagar ← numeroUnidades * 20
        senão se código = 'C'
              então aPagar ← numeroUnidades * 30
              senão início              {CÓDIGO ESTÁ INCORRETO}
                    aPagar ← 0
                    escrever('Código errado')
                    fim
  se aPagar > 0                         {CÓDIGO ERA VÁLIDO}
  então início
        escrever(aPagar)                {INFORMA VALOR A PAGAR}
        se numeroUnidades > 10
        então escrever ('Foram vendidas mais de 10 unidades')
        fim
  fim
```