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

5.4 → comando de repetição condicional `repita/até` por avaliação posterior de condição

O **comando de repetição condicional** por avaliação posterior **`repita/até`** também vincula a execução de um conjunto de comandos ao resultado da avaliação de uma expressão lógica. O comando inicia pela execução do laço e, quando essa execução é concluída, a expressão é avaliada: se o valor lógico obtido for falso, o laço é executado novamente; se for verdadeiro, o comando é encerrado. Isso significa que o laço é sempre executado pelo menos uma vez, independentemente do valor lógico inicial resultante da avaliação da expressão de controle. Observar que, normalmente, o valor inicial da expressão lógica será falso, pois se deseja repetir o laço mais de uma vez. Portanto, é necessário que, em algum momento, o conteúdo de alguma variável utilizada nesta expressão lógica tenha o valor alterado dentro do laço, de forma a modificar o valor resultante de sua avaliação para verdadeiro, evitando assim a ocorrência de um laço – *loop* – infinito.

A sintaxe de um comando de repetição condicional `repita/até` é a seguinte:

```
repita
   <comandos>
até <expressão lógica>
```

Observar que, diferentemente dos comandos anteriores, aqui não é necessário um comando composto, pois a sintaxe aceita múltiplos comandos, delimitados pela cláusula `até`.

O fluxograma representado na Figura 5.5 mostra o funcionamento desse comando, em que o laço de repetição é sempre executado pelo menos uma vez.

O aninhamento de comandos de repetição também se aplica ao comando `repita/até`, incluindo os outros comandos de repetição já vistos.

O Algoritmo 5.7, a seguir, adapta o Algoritmo 5.6, utilizando no laço de repetição um comando `repita/até` em lugar do `enquanto/faça`. Observar que, como o laço desse comando é sempre executado pelo menos uma vez, se tornou necessária a inclusão de um comando condicional logo no início para condicionar a execução do laço ao valor inicial de `nota1`.

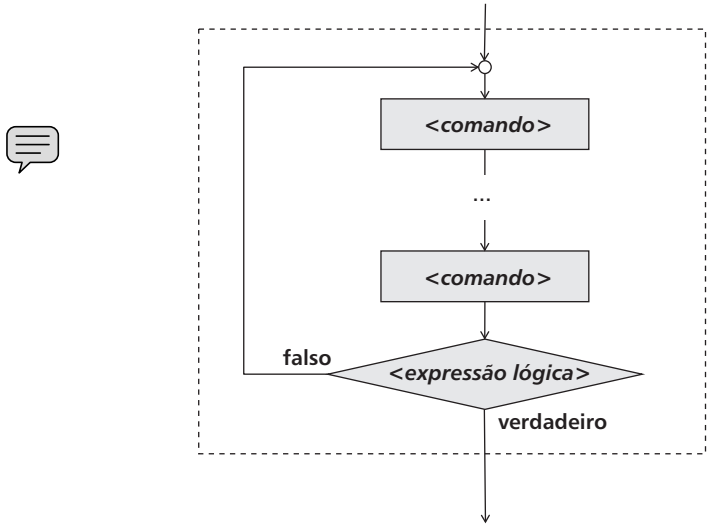

figura 5.5 Fluxograma do comando de repetição `repita/até`.

```
Algoritmo 5.7 - MédiaAlunoETurma_3
{INFORMA MÉDIA DOS ALUNOS DE UMA TURMA E A MÉDIA GERAL DESSA TURMA.
PARA INDICAR FIM DE PROCESSAMENTO, O CONTEÚDO INFORMADO EM NOTA1 SERÁ
NEGATIVO}
  Entradas: nota1, nota2, nota3 (real)
  Saídas: média (real)
          soma_médias (real)
          média_turma (real)
  Variável auxiliar:
          cont_al  (inteiro) {CONTADOR DE ALUNOS PROCESSADOS}
início
  soma_médias ← 0  {SOMA MÉDIAS INDIVIDUAIS: VALOR INICIAL ZERO}
  cont_al ← 0      {CONTADOR DE ALUNOS: VALOR INICIAL ZERO}
  ler(nota1)       {LEITURA DA PRIMEIRA NOTA}
  se nota1 ≥ 0
  então início
        repita
          ler(nota2, nota3)          {LEITURA DAS OUTRAS 2 NOTAS}
          cont_al ← cont_al + 1               {CONTA ALUNO LIDO}
          média  (nota1 + nota2 + nota3) / 3      {CALCULA MÉDIA}
          escrever (cont_al, média)               {INFORMA MÉDIA}
          soma_médias ← soma_médias + média    {SOMA DAS MÉDIAS}
          ler(nota1)   {LEITURA DA PRIMEIRA NOTA DO PRÓXIMO ALUNO}
```

```
        até nota1 < 0                    {FINAL DO COMANDO REPITA}
        fim  {DO COMANDO SE/ENTÃO}
  média_turma ← soma_médias / cont_al          {MÉDIA DA TURMA}
  escrever(média_turma)
fim
```

5.5 → garantia da consistência de dados através de comandos de repetição

Sempre que possível, os valores lidos devem ser verificados quanto à validade antes de seu armazenamento em variáveis. Por exemplo, se o intervalo de valores de notas de provas é de 0 a 10, então qualquer valor de nota informado que não estiver dentro desse intervalo está incorreto e deve ser descartado e substituído por um novo valor válido. A consistência de dados de entrada pode ser implementada através do uso de comandos de repetição condicional. No caso dos algoritmos de processamento de notas incluídos neste capítulo, toda a leitura de nota deveria incluir a consistência do valor informado. Nos exemplos a seguir, a consistência das leituras é garantida através do uso de `enquanto/faça` e de `repita/até`, de acordo com o funcionamento de cada comando.

```
{CONSISTÊNCIA COM COMANDO ENQUANTO/FAÇA}
ler (nota1)
enquanto (nota1 < 0 ou nota1 > 10) e nota1 ≠ -1 faça
  início      {SÓ EXECUTA SE NOTA INVÁLIDA}
  escrever('Nota inválida! Informe novamente. ')
  ler(nota1)
  fim {ENQUANTO}
{CONSISTÊNCIA DE DADOS COM COMANDO REPITA/ATÉ}
repita    {CONSISTÊNCIA DE NOTA2}
  ler(nota2)  {SEMPRE EXECUTA 1 VEZ, REPETE SE INVÁLIDA}
  se nota2 < 0 ou nota2 > 10
  então escrever('Nota inválida! Informe novamente.')
até (nota2 ≥ 0 e nota2 ≤ 10)
```

Apesar das diferenças no modo de funcionamento, os comandos `enquanto/faça` e `repita/até` podem ser utilizados indistintamente nas situações em que o uso de comandos desse tipo for adequado. Entretanto, o comando `repita/até`, por apresentar a característica de sempre executar o conteúdo do laço pelo menos uma vez, é o mais indicado para a consistência de dados de entrada. A leitura dos dados deverá ocorrer obrigatoriamente pelo menos uma vez, e a avaliação do resultado da leitura determinará o encerramento (dado lido válido) ou a repetição do comando (dado lido inválido).

O trecho a seguir apresenta a utilização do comando `repita/até` para a consistência da leitura de uma letra minúscula, garantindo que o caractere digitado esteja dentro do intervalo "a" a "z", mas sem emitir mensagem de erro de digitação.

```
repita
  escrever('Informe uma letra minúscula:')
  ler(letra)
até (letra ≥ 'a' e letra ≤ 'z') {SE VALOR VÁLIDO, NÃO REPETE}
```

5.6 selecionando o comando de repetição mais adequado

As normas de bom estilo de programação recomendam que os comandos de repetição tenham seu início e fim e suas condições de repetição claramente explicitados.

Havendo mais de um comando de repetição possível para atender uma particular situação, a escolha deve recair sobre o comando com as características mais adequadas ao caso em análise.

Se a repetição implementada for por contagem ou pela variação de conteúdo de uma variável, através de incrementos constantes e definidos em um intervalo previamente conhecido, então o comando de repetição indicado é o `para/faça`. Nesse caso, a variável de controle, a condição de repetição e o incremento aplicado após cada execução do laço devem estar claramente definidos no cabeçalho do comando `para/faça` e devem ser evitadas alterações desses elementos no corpo do laço.

Os comandos de repetição condicional são indicados para a solução de problemas em que o encerramento das repetições está relacionado à detecção de uma condição cuja ocorrência não pode ser predeterminada.

Se existe a possibilidade do bloco de repetições não precisar ser executado nem uma vez, então o uso do comando `enquanto/faça` é indicado. Um exemplo dessa opção é o processamento de pedidos de uma empresa durante o período comercial: nesse caso, é recomendável que o usuário seja informado da eventual inexistência de processamento, caso nenhum pedido ocorra no período em questão.

Se o bloco de repetições for sempre executado pelo menos uma vez, então a opção deve recair no comando `repita/até`. Um exemplo do uso adequado do `repita/até` é o processamento que inclui *menus* de opções, uma delas sendo o encerramento da execução: nesse caso, sempre ocorrerá a leitura e processamento de pelo menos uma das opções do *menu*.