O Instruções rotuladas break e continue

O.1 Introdução

No Capítulo 5, discutimos as instruções `break` e `continue` do Java que permitem aos programadores alterar o fluxo de controle das instruções de controle. O Java também fornece as instruções `break` e `continue` rotuladas para situações em que um programador precisa alterar convenientemente o fluxo de controle nas instruções de controle aninhadas. Este apêndice demonstra as instruções `break` e `continue` rotuladas com exemplos que utilizam instruções `for` aninhadas.

O.2 Instrução rotulada break

A instrução `break` apresentada na Seção 5.7 permite a um programa sair das instruções `while`, `for`, `do...while` ou `switch` nas quais a instrução `break` aparece. Às vezes, essas instruções de controle são aninhadas em outras instruções de repetição. Um programa precisaria sair da instrução de controle aninhada inteira em uma única operação, em vez de esperar que ele complete a execução normalmente. Para sair dessas instruções de controle aninhadas, você pode utilizar a **instrução `break` rotulada**. Essa instrução, quando executada em uma `while`, `for`, `do...while` ou `switch`, resulta em uma saída imediata dessa instrução de controle e de quaisquer instruções envolventes. A execução do programa é retomada com a primeira instrução depois da **instrução rotulada** envolvente. A instrução que segue o rótulo pode ser uma instrução de repetição ou um bloco no qual uma instrução de repetição aparece. A Figura O.1 demonstra a instrução `break` rotulada em uma instrução `for` aninhada.

O bloco (linhas 7–26 na Figura O.1) inicia com um **rótulo** (um identificador seguido por um dois-pontos) na linha 7; aqui utilizamos o rótulo `stop:` O bloco é colocado entre chaves (linhas 8 e 26) e inclui a `for` aninhada (linhas 10–22) e a instrução de saída na linha 25. Quando a instrução `if` na linha 15 detecta que `row` é igual a `5`, a instrução `break` na linha 16 executa. Essa instrução termina as duas `for` nas linhas 13–19 e sua `for` envolvente nas linhas 10–22. Em seguida, o programa prossegue imediatamente para a primeira instrução depois do bloco rotulado — nesse caso, o fim do método main é alcançado e o programa termina. A `for` externa executa completamente o seu corpo somente quatro vezes. A instrução de saída na linha 25 nunca executa, porque ela está no corpo do bloco rotulado e a `for` externa nunca completa.

Boa prática de programação O.1

Muitos níveis de instruções de controle aninhadas podem tornar um programa difícil de ler. Como uma regra geral, tente evitar a utilização de mais de três níveis de aninhamento.

```
1  Figura. O.1 5.13 BreakLabelTest.java
2  // Instrução break rotulada saindo de uma instrução for aninhada.
3  public class BreakLabelTest
4  {
5     public static void main( String[] args )
6     {
7        stop: // bloco rotulado
```

```
        {
           // conta 10 linhas
           for ( int row = 1; row <= 10; row++ )
           {
              // conta 5 colunas
              for ( int column = 1; column <= 5 ; column++ )
              {
                 if ( row == 5 ) // se a linha for 5,
                    break stop;  // pula para o final do bloco de parada

                 System.out.print( "* " );
              } // fim do for interno

              System.out.println(); // gera a saída de nova linha
           } // fim do for externo

           // a linha seguinte é pulada
           System.out.println( "\nLoops terminated normally" );
        } // fim do bloco rotulado
     } // fim de main
  } // fim da classe BreakLabelTest
```

```
* * * * *
* * * * *
* * * * *
* * * * *
```

Figurara. O.1 | A instrução `break` rotulada saindo de uma instrução `for` aninhada.

O.3 Instrução rotulada `continue`

A instrução `continue` apresentada na Seção 5.7 prossegue para a próxima iteração (repetição) da `while`, `for` ou `do...while` imediatamente envolvente. A **instrução `continue` rotulada** pula as instruções restantes no corpo dessa instrução e quaisquer números de instruções de repetição envolventes e prossegue para a próxima iteração da **instrução de repetição rotulada** envolvente (isto é, uma `for`, `while` ou `do...while` precedida por um rótulo). Nas instruções `while` e `do...while` rotuladas, o programa avalia o teste de continuação de loop do loop rotulado logo após que a instrução `continue` executa. Em uma `for` rotulada, a expressão de incremento é executada e o teste do loop de continuação é avaliado. A Figura O.2 utiliza uma instrução `continue` rotulada em uma `for` aninhada para permitir que a execução continue para a próxima iteração da `for` externa.

```
// Figura. O.2: ContinueLabelTest.java
// Instrução continue rotulada terminando uma instrução for aninhada.
public class ContinueLabelTest
{
   public static void main( String[] args )
   {
      nextRow: // rótulo-alvo da instrução continue

         // conta 5 linhas
         for ( int row = 1; row <= 5; row++ )
         {
            System.out.println(); // gera a saída de nova linha

            // conta 10 colunas por linha
            for ( int column = 1; column <= 10; column++ )
            {
               // se a coluna for maior que a linha, inicia a próxima linha
               if ( column > row )
                  continue nextRow; // próxima iteração do loop rotulado

               System.out.print( "* " );
            } // fim do for interno
         } // fim do for externo

      System.out.println(); // gera a saída de nova linha
   } // fim de main
} // fim da classe ContinueLabelTest
```

```
*
* *
* * *
* * * *
* * * * *
```

Figurara. O.2 | Instrução `continue` rotulada terminando uma instrução `for` aninhada.

A `for` rotulada (linhas 7–23), na verdade, inicia no rótulo `nextRow`. Quando a instrução `if` na linha 18 da `for` interna (linhas 15–22) detecta que `column` é maior do que `row`, a instrução `continue` na linha 19 executa e o controle do programa continua com o incremento da variável de controle `row` do loop `for` externo. Mesmo que a `for` interna conte de 1 a 10, o número de caracteres de saída `*` em uma linha nunca excede ao valor de `row`, criando um padrão triangular interessante.