

5.5 do...while Iteration Statement

The `do...while` **iteration statement** is similar to the `while` statement. A `while` tests its loop-continuation condition at the *beginning* of the loop, *before* executing the loop's body; if the condition is `false`, the body *never* executes. A `do...while` tests its loop-continuation condition *after* executing the loop's body; therefore, *the body always executes at least once*. When a `do...while` statement terminates, execution continues with the next statement in sequence. [Figure 5.7](#) uses a `do...while` to output the numbers 1–10.

```
1  // Fig. 5.7: DoWhileTest.java
2  // do...while iteration statement.
3
4  public class DoWhileTest {
5      public static void main(String[] args) {
6          int counter = 1;
7
8          do {
9              System.out.printf("%d ", counter);
10             ++counter;
11         } while (counter <= 10);
12
13         System.out.println();
14     }
15 }
```

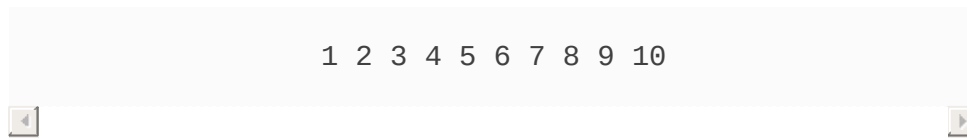


Fig. 5.7

do...while iteration statement.

Line 6 declares and initializes control variable `counter`. Upon entering the `do...while` statement, line 9 outputs `counter`'s value and line 10 increments `counter`. Then the program evaluates the loop-continuation test at the *bottom* of the loop (line 11). If the condition is `true`, the loop continues at the first body statement (line 9). If the condition is `false`, the loop terminates and the program continues at the next statement after the loop.

UML Activity Diagram for the `do...while` Iteration Statement

Figure 5.8 contains the UML activity diagram for the `do...while` statement. This diagram makes it clear that the loop-continuation condition is not evaluated until *after* the loop performs the action state *at least once*. Compare this activity diagram with that of the `while` statement (Fig. 4.6).

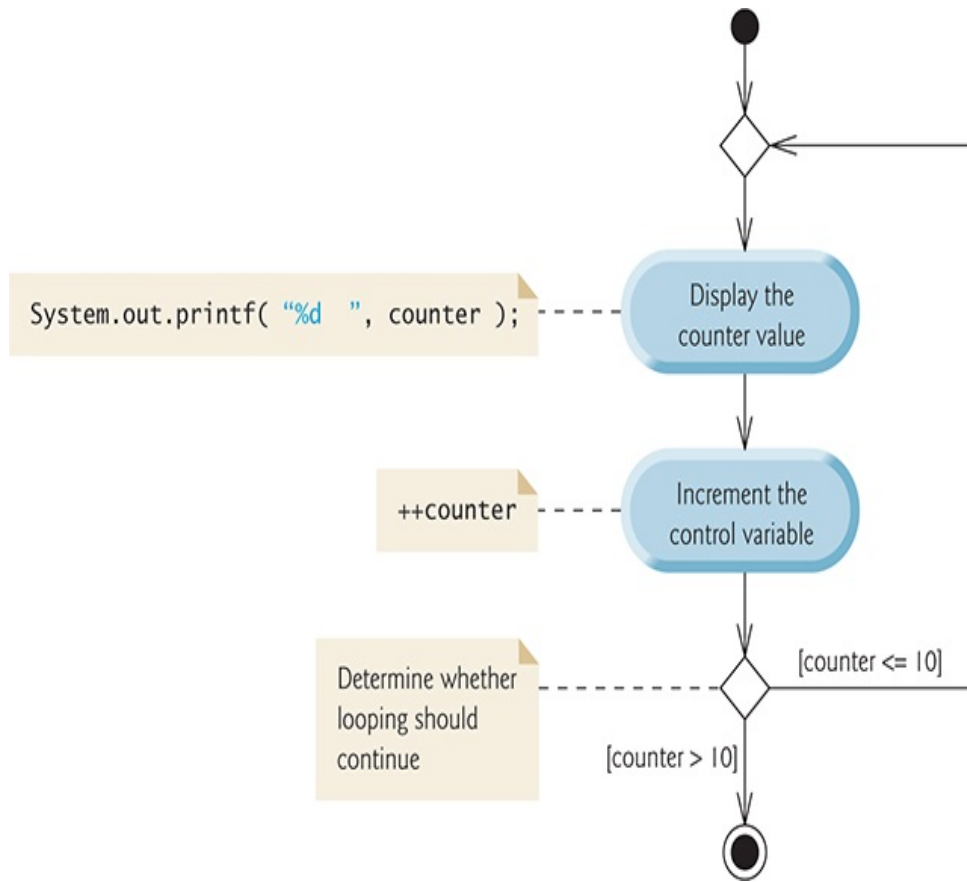


Fig. 5.8

do...while iteration statement UML activity diagram.

Description