5.2 Essentials of Counter-Controlled Iteration

This section uses the `while` iteration statement introduced in Chapter 4 to formalize the elements required to perform counter-controlled iteration, which requires

1. a **control variable** (or loop counter)
2. the **initial value** of the control variable
3. the **increment** by which the control variable is modified each time through the loop (also known as **each iteration of the loop**)
4. the **loop-continuation condition** that determines if looping should continue.

To see these elements of counter-controlled iteration, consider the application of Fig. 5.1, which uses a loop to display the numbers from 1 through 10.

```
1 // Fig. 5.1: WhileCounter.java
2 // Counter-controlled iteration with the while ite
3
4 public class WhileCounter {
5    public static void main(String[] args) {
6       int counter = 1; // declare and initialize c
7
8       while (counter <= 10) { // loop-continuation
9          System.out.printf("%d ", counter);
10         ++counter; // increment control variable
11       }
```

```
12
13       System.out.println();
14    }
15 }
```

```
1 2 3 4 5 6 7 8 9 10
```

Fig. 5.1

Counter-controlled iteration with the `while` iteration statement.

In Fig. 5.1, the elements of counter-controlled iteration are defined in lines 6, 8 and 10. Line 6 *declares* the control variable (`counter`) as an `int`, *reserves space* for it in memory and sets its *initial value* to `1`. Variable `counter` also could have been declared and initialized with the following local-variable declaration and assignment statements:

```
int counter; // declare counter
counter = 1; // initialize counter to 1
```

Line 9 displays control variable `counter`'s value during each iteration of the loop. Line 10 *increments* the control variable by 1 for each iteration of the loop. The loop-continuation condition in the `while` (line 8) tests whether the value of the control variable is less than or equal to `10` (the final value for

which the condition is `true`). The program performs the body of this `while` even when the control variable is `10`. The loop terminates when the control variable exceeds `10` (i.e., `counter` becomes `11`).

Common Programming Error 5.1

Because floating-point values may be approximate, controlling loops with floating-point variables may result in imprecise counter values and inaccurate termination tests.

Error-Prevention Tip 5.1

Use integers to control counting loops.