

Flow of Control: Loops

Chapter 4

Objectives

- Design a loop
- Use while, do, and for in a program
- Use the for-each with enumerations
- Use assertion checks
- Use repetition in a graphics program
- Use drawString to display text in a graphics program

Java Loop Statements: Outline

- The while statement
- The do-while statement
- The for Statement

Java Loop Statements

- A portion of a program that repeats a statement or a group of statements is called a *loop*.
- The statement or group of statements to be repeated is called the *body* of the loop.
- A loop could be used to compute grades for each student in a class.
- There must be a means of exiting the loop.

- Also called a while loop
- A while statement repeats while a controlling boolean expression remains true
- The loop body typically contains an action that ultimately causes the controlling boolean expression to become false.

• View <u>sample program</u>, Listing 4.1 class WhileDemo



 Figure 4.1
 The action of the while loop in Listing 4.1



```
• Syntax
  while (Boolean Expression)
     Body_Statement
  or
  while (Boolean Expression)
     First Statement
     Second Statement
     ...
```

Figure 4.2 Semantics of the while statement



- Also called a do-while loop
- Similar to a while statement, except that the loop body is executed at least once
- Syntax

do

Body_Statement
while (Boolean_Expression);

Don't forget the semicolon!

• View <u>sample program</u>, listing 4.2 class DoWhileDemo

Enter a number: 2	
1, 2, Buckle my shoe. Enter a number: 3 1, 2, 3, Buckle my shoe	
Buckle my shoe.	Sample screen output
D D D D D D D D D D D D D D	ys ce.

do

{

• Figure 4.3 The Action of the do-while Loop in Listing 4.2



- First, the loop body is executed.
- Then the boolean expression is checked.
 - As long as it is true, the loop is executed again.
 - If it is false, the loop is exited.
- Equivalent while statement
 Statement(s)_S1
 while (Boolean_Condition)
 Statement(s) S1

 Figure 4.4 The Semantics of the do-while Statement



- Given
 - Volume a roach: 0.002 cubic feet
 - Starting roach population
 - Rate of increase: 95%/week
 - Volume of a house
- Find
 - Number of weeks to exceed the capacity of the house
 - Number and volume of roaches

Algorithm for roach population program (rough draft)

- 1. Get volume of house.
- 2. Get initial number of roaches in house.
- 3. Compute number of weeks until the house is full of roaches.
- 4. Display results.

Variables Needed

GROWTH_RATE —weekly growth rate of the roach population (a constant 0.95)

ONE_BUG_VOLUME —volume of an average roach (a constant 0.002)

houseVolume — volume of the house

startPopulation —initial number of roaches ctd. ...

Variables Needed countWeeks —week counter Population —current number of roaches totalBugVolume —total volume of all the roaches newBugs —number of roaches hatched this week newBugVolume —volume of new roaches

- View more <u>detailed algorithm</u>
- View <u>sample program</u>, listing 4.3 class BugTrouble

Enter the total volume of your house in cubic feet: 20000 Enter the estimated number of roaches in your house: 100 Starting with a roach population of 100 and a house with a volume of 20000.0 cubic feet, after 18 weeks, the house will be filled with 16619693 roaches. They will fill a volume of 33239 cubic feet. Better call Debugging Experts Inc. Sample screen output

Infinite Loops

- A loop which repeats without ever ending is called an *infinite loop*.
- If the controlling boolean expression never becomes false, a while loop or a do-while loop will repeat without ending.
- A negative growth rate in the preceding problem causes totalBugVolume always to be less than houseVolume, so that the loop never ends.

Nested Loops

- The body of a loop can contain any kind of statements, including another loop.
- In the previous example
 - The average score was computed using a while loop.
 - This while loop was placed inside a dowhile loop so the process could be repeated for other sets of exam scores.

Nested Loops

• View <u>sample program</u>, listing 4.4 class ExamAverager

> Want to average another exam? Enter yes or no. yes Enter all the scores to be averaged. Enter a negative number after you have entered all the scores. 90 70 80 -1 The average is 80.0 Want to average another exam? Enter yes or no.

Sample screen output

- A **for** statement executes the body of a loop a fixed number of times.
- Example

for (count = 1; count < 3; count++)
System.out.println(count);</pre>

• Syntax

for (Initialization, Condition, Update)
 Body_Statement

- Body_Statement can be either a simple statement or a compound statement in { }.
- Corresponding while statement
 Initialization
 while (Condition)
 Body Statement Including Update

• View sample program, Listing 4.4 class ForDemo

3 and counting. 2 and counting. 1 and counting. 0 and counting. Blast off!

Sample screen output

 Figure 4.5 The action of the for loop in listing 4.5



```
for (countDown = 3; countDown >= 0; countDown--)
{
    System.out.println(countDown);
    System.out.println("and counting.");
}
```

 Figure 4.6 The semantics of the for statement



 Possible to declare variables within a for statement

Note that variable n is local to the loop

- A comma separates multiple initializations
- Example
 for (n = 1, product = 1; n <= 10; n++)
 product = product * n;</pre>
- Only one boolean expression is allowed, but it can consist of &&s, ||s, and !s.
- Multiple update actions are allowed, too.
 for (n = 1, product = 1; n <= 10; product = product * n, n++);

The for-each Statement

- Possible to step through values of an enumeration type
- Example

enum Suit {CLUBS, DIAMONDS, HEARTS, SPADES}
for (Suit nextSuit : Suit.values())
System.out.print(nextSuit + " ");
System.out.println();

Programming with Loops: Outline

- The Loop Body
- Initializing Statements
- Controlling Loop Iterations
- break and continue statements
- Loop Bugs
- Tracing Variables
- Assertion checks

The Loop Body

- To design the loop body, write out the actions the code must accomplish.
- Then look for a repeated pattern.
 - The pattern need not start with the first action.
 - The repeated pattern will form the body of the loop.
 - Some actions may need to be done after the pattern stops repeating.

Initializing Statements

- Some variables need to have a value before the loop begins.
 - Sometimes this is determined by what is supposed to happen after one loop iteration.
 - Often variables have an initial value of zero or one, but not always.
- Other variables get values only while the loop is iterating.

- If the number of iterations is known before the loop starts, the loop is called a *count-controlled loop*.
 - Use a for loop.
- Asking the user before each iteration if it is time to end the loop is called the *ask-before-iterating technique*.
 - Appropriate for a small number of iterations
 - Use a while loop or a do-while loop.

- For large input lists, a *sentinel value* can be used to signal the end of the list.
 - The sentinel value must be different from all the other possible inputs.
 - A negative number following a long list of nonnegative exam scores could be suitable.

 Example - reading a list of scores followed by a sentinel value int next = keyboard.nextInt();

```
while (next >= 0)
```

{

}

```
Process_The_Score
next = keyboard.nextInt();
```

- Using a boolean variable to end the loop
- View <u>sample program</u>, listing 4.6 class BooleanDemo

Enter nonnegative numbers. Place a negative number at the end to serve as an end marker. 1 2 3 -1 The sum of the numbers is 6 Sample screen output

Programming Example

- Spending Spree
 - You have \$100 to spend in a store
 - Maximum 3 items
 - Computer tracks spending and item count
 - When item chosen, computer tells you whether or not you can buy it
- Client wants adaptable program
 - Able to change amount and maximum number of items
- View <u>sample algorithm</u>

Programming Example

• View <u>sample program</u>, listing 4.7 class <u>SpendingSpree</u>

> You may buy up to 3 items costing no more than \$100. Enter cost of item #1: \$80 You may buy this item. You spent \$80 so far. You may buy up to 2 items costing no more than \$20. Enter cost of item #2: \$20 You may buy this item. You spent \$100 so far. You are out of money. You spent \$100, and are done shopping.

Sample screen output

The break Statement in Loops

- A break statement can be used to end a loop immediately.
- The break statement ends only the innermost loop or switch statement that contains the break statement.
- **break** statements make loops more difficult to understand.
- Use **break** statements sparingly (if ever).

The break Statement in Loops

 Note program fragment, ending a loop with a break statement, listing 4.8



The continue Statement in Loops

- A continue statement
 - Ends current loop iteration
 - Begins the next one
- Text recommends avoiding use
 - Introduce unneeded complications

Tracing Variables

- *Tracing variables* means watching the variables change while the program is running.
 - Simply insert temporary output statements in your program to print of the values of variables of interest
 - Or, learn to use the debugging facility that may be provided by your system.

Assertion Checks

- Assertion : something that says something about the state of the program
 - Can be true or false
 - Should be true when no mistakes in running program

Assertion Checks

Example found in comments

```
//n == 1
while (n < limit)
{
n = 2 * n;
}
//n >= limit
//n is the smallest power of 2 >= limit
```

• Syntax for assertion check Assert Boolean Expression;

Assertion Checks

Equivalent example using assert

```
assert n == 1;
while (n < limit)
{
  n = 2 * n;
}
assert n >= limit;
//n is the smallest power of 2 >= limit.
```

Loop Bugs

- Common loop bugs
 - Unintended infinite loops
 - Off-by-one errors
 - Testing equality of floating-point numbers
- Subtle infinite loops
 - The loop may terminate for some input values, but not for others.
 - For example, you can't get out of debt when the monthly penalty exceeds the monthly payment.

Programming Example

- A multiface Applet
 - Uses loop to draw several smiley faces
 - Uses if statement to alter appearance
- View <u>sample program</u>, listing 4.9 class MultipleFaces

Programming Example



The drawString Method

- Similar to methods for drawing ovals
- Displays text
- Example canvas.drawString("Hello",10, 20);
 - Writes word Hello at point (10, 20)
- Used to place "Kiss, Kiss" and "Tee Hee" on screen in listing 4.9

Summary

- A loop is a programming construct that repeats an action
- Java has the while, the do-while, and the for statements
- The while and do-while repeat the loop while a condition is true
- The logic of a for statement is identical to the while

Summary

- Loops may be ended using a sentinel value or a boolean value
- Typical loop bugs include infinite loops or loops which are off by 1 iteration
- Variables may be traced by including temporary output statements or a debugging utility
- The **assert** statement can be used to check conditions at run time
- Use drawString to display text in an applet