
JAVA™
AN INTRODUCTION TO
PROBLEM SOLVING
AND PROGRAMMING
WALTER SAVITCH

FOR LOOP

The `for` Statement

- A `for` statement executes the body of a loop a fixed number of times.
- Examples

```
for (count = 1; count < 3; count++)
    System.out.println(count);
```

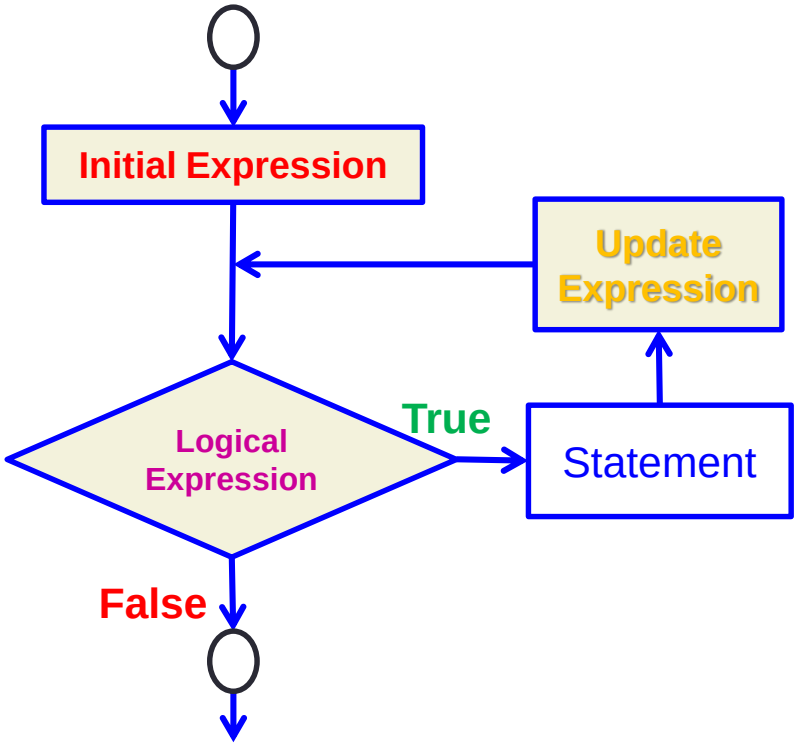
The `for` Statement

SYNTAX

```
for (initial expression; logical expression; update expression)
   statement;   //loop body: the statement to be repeated
```

Example

```
for (counter = 1; counter<=N; counter++)
   sum += counter;   //loop body
```



Execution:

1. Initial statement(s) executes.
2. Loop condition is evaluated.
3. If loop condition is true,
   i. execute `for` loop statement (loop body)
   ii. execute update statement(s).
   iii. Go back to step 2
4. If loop condition is false continue with remaining statements

Counter-controlled loops

- The for loop is a specialized form of a while loop.
- It's primary purpose is to simplify the writing of counter-controlled loops.
- The for loop is typically called a **counted** or **indexed** for loop.
- If there's more than one statement in the body, use a block {}

- While loop

```
N = …
counter = 1
Loop while (counter<=N)
    .
    .
    counter = counter + 1
End loop
```

```
N = …
For(counter = 1; counter <= N; counter=counter+ 1)
    .
    .
End For
```

Initialization **Condition** **Increment \ Decrement**

while VS for

while

```
int x = 0, i = 1;

while (i < 4)
{   x = x + i;
    i++;
}

System.out.println(x);
System.out.println(i);
```

for

```
int x = 0, i;

for (i = 1; i < 4; i++)
{   x = x + i;
}


System.out.println(x);
System.out.println(i);
```

while VS for

while

```
int x = 0, i = 1;

while (i < 4)
{   x = x + i;
    i++;
}

System.out.println(x);
System.out.println(i);
```

for

```
int x = 0, i;

for (i = 1; i < 4; i++)
{   x = x + i;
    i++;
}

Sy   em.out.println(x);
     tem.out.println(i);
```

What would happen if we left this in here?

And what if we changed it to `i--;`?

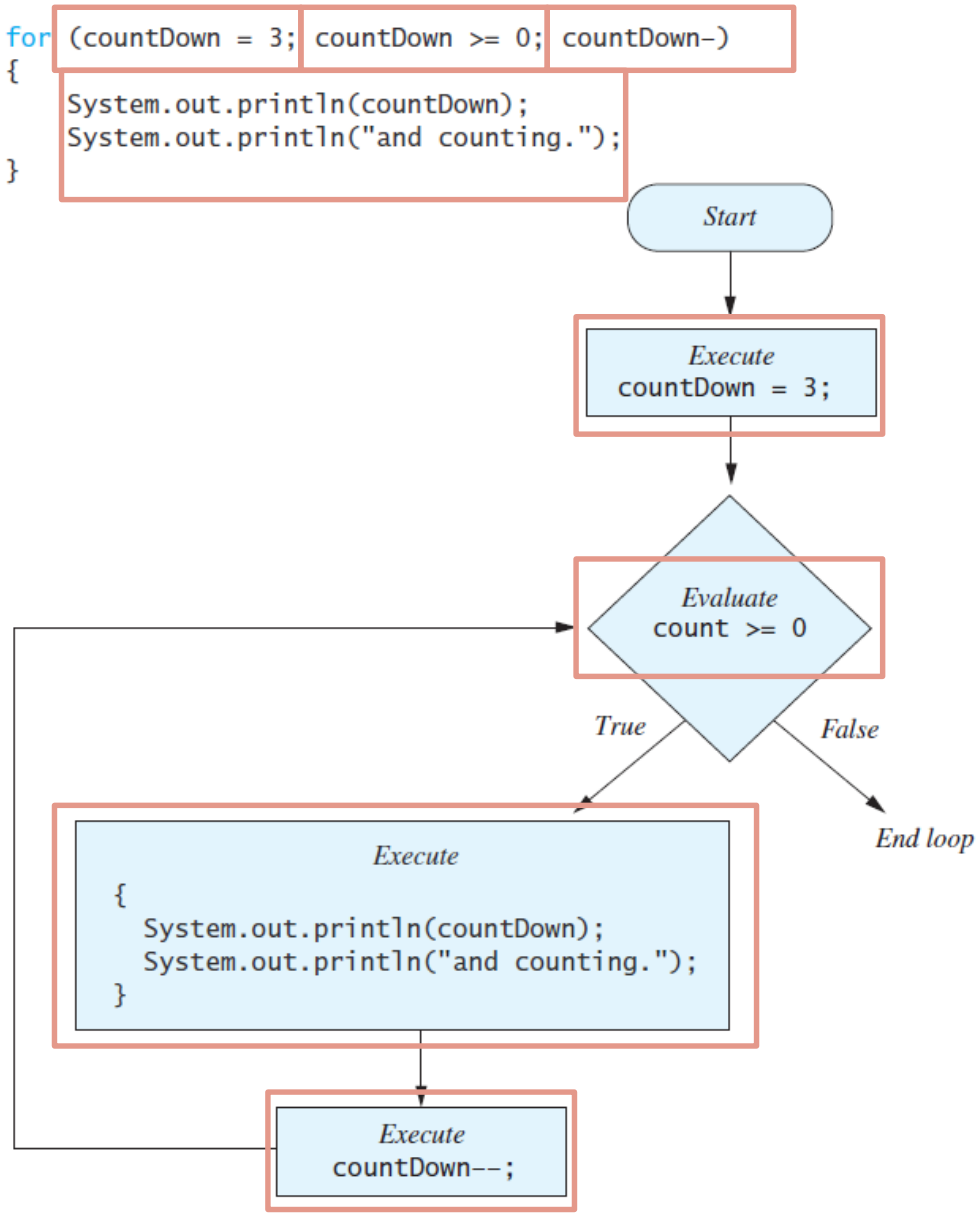
sample program, listing 4.5

```
public class ForDemo
{
    public static void main (String [] args)
    {
        int countDown;
        for (countDown = 3 ; countDown >= 0 ; countDown--)
        {
            System.out.println (countDown);
            System.out.println ("and counting.");
        }
        System.out.println ("Blast off!");
    }
}
```

```
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



The `for` Statement

What is the difference between these two for loops?

```
for (i = 1; i <= 5; i++)
{   System.out.println("Hello");
    System.out.println("*");
}
```

1. This for loop outputs the word Hello and a star (on separate lines) five times

```
for (i = 1; i <= 5; i++)
   System.out.println("Hello");
   System.out.println("*");
```

2. This for loop outputs the word `Hello` five times and the star only once

Programming Example: Square

Write a program that reads a set of 1000 integers and prints the square of each integer.

INPUT

A set of int numbers (variable: number, type: int)
N = 1000 (type: int)

OUTPUT

The square of each read number (variable: square, type: int)

PROCESS

```
Repeat N times:
        read number
        square = number * number
        print square
```

Programming Example: Square

```
1  // import necessary libraries
2  import java.util.*;
3  public class forLoop
4  {
5      static final int N = 1000;            //constant declaration
6     // instantiate the object read from the class Scanner
7     static Scanner read  = new Scanner (System.in);
8     public static void main (String[] args)
9       {
10        // Declaration section: to declare needed variables
11           int number, square, counter;
12           for (counter = 0; counter < N; counter++)
13           {
14              // Input section: to enter values of used variables
15              System.out.println ("Enter an integer number");
16              number = read.nextInt();
17            // Processing section: processing statements
18              square = number * number;
19           // Output section: display program output
20              System.out.println ("Square = " + square);
21           } //end for loop
22      } // end main
23 } // end class
```

Modify the program to read from the user the number of integers

counter is int

Increasing step ➔ final value (N=1000) > initial value (0)

Programming Example: Classify Numbers

- Input:

  `N` integers (positive, negative, and zeros).

```
int N = 20;  //N easily modified
```

- Output:

  Number of 0s,

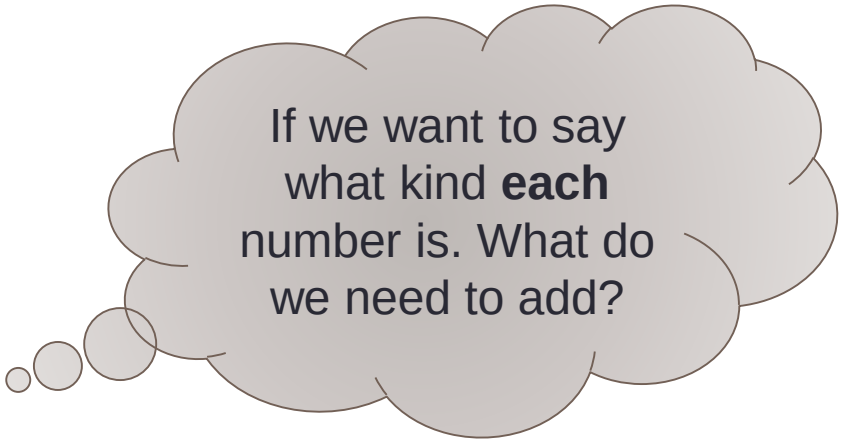
  number of **even** integers,

  number of **odd** integers.

- Processing ??

Programming Example: Classify Numbers

```
int N = 20, number, evens = 0, odds=0, zeros = 0;
System.out.println("Enter " + N + " integers:");
for (int counter = 1; counter <= N; counter++)
{     number = console.nextInt();

      switch (number % 2)
      {    case 0: evens++;

                   if (number == 0)
                        zeros++;


                   break;
           case 1:
           case -1: odds++;

       } //end switch

} //end for loop
System.out.printf("%d evens and %d odds and %d zeros\n",
                    evens, odds, zeros);
```



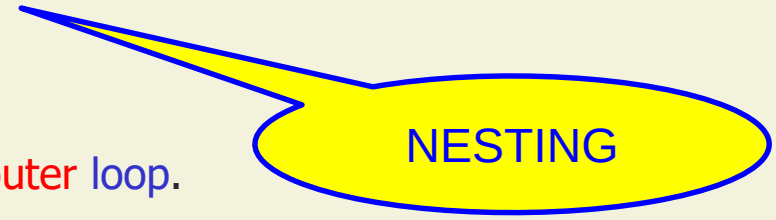
Programming Example: Classify Numbers

```
int N = 20, number, evens = 0, odds=0, zeros = 0;
System.out.println("Enter " + N + " integers:");
for (int counter = 1; counter <= N; counter++)
{    number = console.nextInt();
     System.out.print(number);
     switch (number % 2)
     {   case 0: evens++;
                 System.out.print(" is even");
                 if (number == 0)
                 {    zeros++;
                      System.out.print(" and a zero");
                 }
                 break;
         case 1:
         case -1: odds++;
                  System.out.print(" is odd");
      } //end switch
      System.out.println();
} //end for loop
System.out.printf("%d evens and %d odds and %d zeros\n",
                  evens, odds, zeros);
```

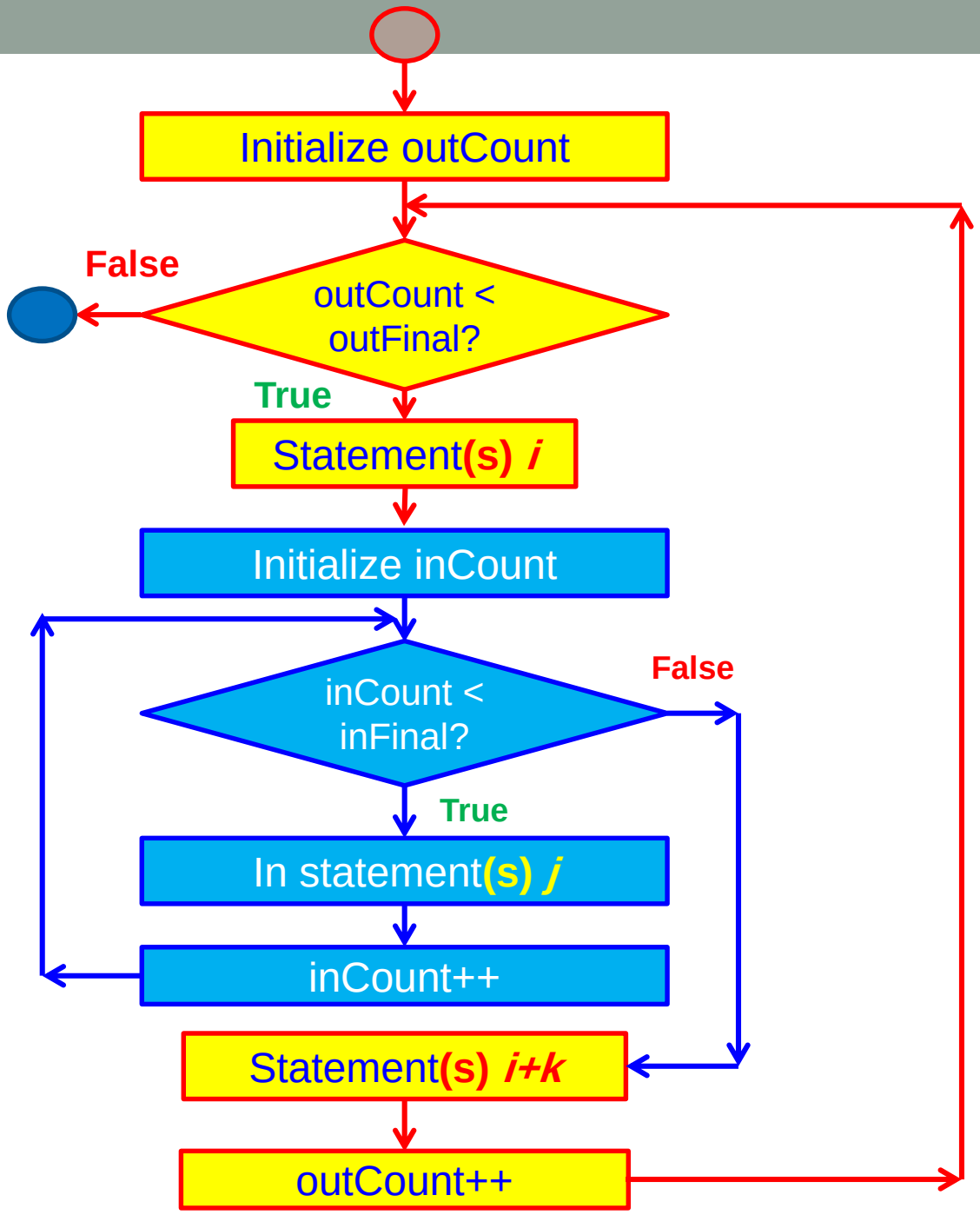
Nested `for` - Syntax

SYNTAX

```
for (initial expression 1; logical expression 1; update expression 1)
{
    statement i;
    statement i+1;
    for (initial expression 2; logical expression 2; update expression 2)
    {
            statement j;
            statement j+1;        // the inner loop.
            …
            statement m;
    }
    statement i+k;
    …                             // the outer loop.
    statement n;
}
```



- Each loop (inner and outer) has its own counter.


Initialize outCount
False
outCount < outFinal?
True
Statement(s) i
Initialize inCount
inCount < inFinal?
False
True
In statement(s) j
inCount++
Statement(s) i+k
outCount++

Nested `for` – Example 1

Example 1

```
1  int outCount, inCount = 0;
2  System.out.println (“Start the loops”);
3  for (outCount = 0; outCount < 2; outCount++)
4      {
5        System.out.printf (“Outer = %3d%n”, outCount);
6        for (inCount = 0; inCount < 3; inCount++)
7              System.out.printf (“\tInner = %3d ”, inCount);
8        System.out.println(“\n”);
9      } //end for outCount
10 System.out.printf (“After the outer loop ends, outCount = %d,
11 inCount = %d”, outCount, inCount);
```

Output

```
1  Start the loops
2  Outer = ~~0
3                Inner = ~~0 Inner = ~~1 Inner = ~~2
4
5  Outer = ~~1
6                Inner = ~~0 Inner = ~~1 Inner = ~~2
7
8  After the outer loop ends, outCount = 2, inCount = 3
```

Nested `for` – Example 2 – analysis

Write a program that produces the following output:

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

Line number	Number of stars
1	1
2	2
3	3
4	4
5	5

If the line number (line) is the outer counter

➔ Initial value =1, final value: line <=5, step: line++

If the number of the stars (stars) is the inner counter

➔ Initial value = 1, final value: stars <= line, step: star++

Nested `for` – Example 2 – code

```
1  public class nestedFor
2  {
3     static final char ASTERISK = '*';
4     public static void main (String[] args)
5        {
6           // Declaration section: to declare needed variables
7              int line, stars;                    //loop counters
8           // Processing section: processing statements
9           // Output section: display program output
10             for (line = 1; line <= 5; line++)
11                {
12                for (stars = 1; stars <= line; stars++)
13                      System.out.print (ASTERISK);
14                 System.out.print ("\n");
15                } //end for (line =…
16       } // end main
17 } // end class
```

Nested `for` – Example 3

```
//What does this code do?
for (i = 1; i <= 5; i++)
{
      for (j = 1; j <= 10; j++)
            System.out.printf("%3d", i*j);

       System.out.println();
}
```

```
Output

  1   2   3   4   5   6   7   8   9  10
  2   4   6   8  10  12  14  16  18  20
  3   6   9  12  15  18  21  24  27  30
  4   8  12  16  20  24  28  32  36  40
  5  10  15  20  25  30  35  40  45  50
```