

**Flow of Control:  
Loops** **4**

## LISTING 4.1 A while Loop

---

```
import java.util.Scanner;
public class WhileDemo
{
    public static void main(String[] args)
    {
        int count, number;

        System.out.println("Enter a number");
        Scanner keyboard = new Scanner(System.in);
        number = keyboard.nextInt();

        count = 1;
        while (count <= number)
        {
            System.out.print(count + ", ");
            count++;
        }

        System.out.println();
        System.out.println("Buckle my shoe.");
    }
}
```

---

---

### Sample Screen Output 1

```
Enter a number:  
2  
1, 2,  
Buckle my shoe.
```

---

### Sample Screen Output 2

```
Enter a number:  
3  
1, 2, 3,  
Buckle my shoe.
```

---

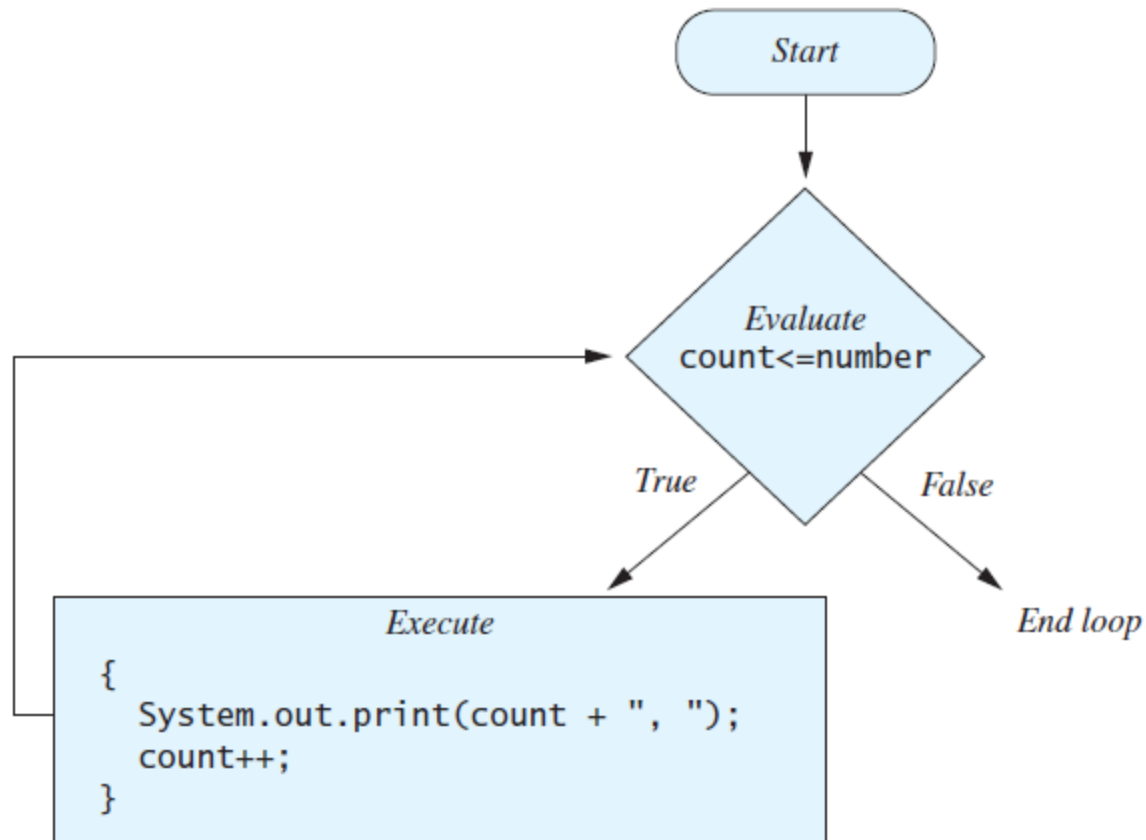
### Sample Screen Output 3

```
Enter a number:  
0  
←  
Buckle my shoe.
```

*The loop body is  
iterated zero times.*

**FIGURE 4.1** The Action of the `while` Loop in Listing 4.1

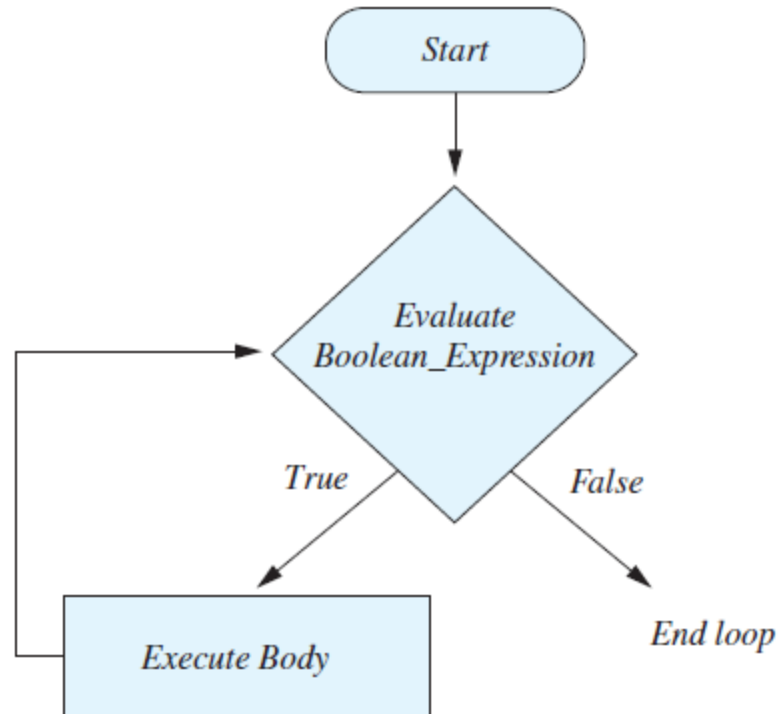
```
while (count <= number)
{
    System.out.print(count + ", ");
    count++;
}
```



**FIGURE 4.2** The Semantics of the `while` Statement

---

`while` (*Boolean\_Expression*)  
*Body*



## LISTING 4.2 A do-while Loop

---

```
import java.util.Scanner;
public class DoWhileDemo
{
    public static void main(String[] args)
    {
        int count, number;
        System.out.println("Enter a number");
        Scanner keyboard = new Scanner(System.in);
        number = keyboard.nextInt();

        count = 1;
        do
        {
            System.out.print(count + ", ");
            count++;
        } while (count <= number);

        System.out.println();
        System.out.println("Buckle my shoe.");
    }
}
```

---

---

### Sample Screen Output 1

```
Enter a number:  
2  
1, 2,  
Buckle my shoe.
```

---

### Sample Screen Output 2

```
Enter a number:  
3  
1, 2, 3,  
Buckle my shoe.
```

---

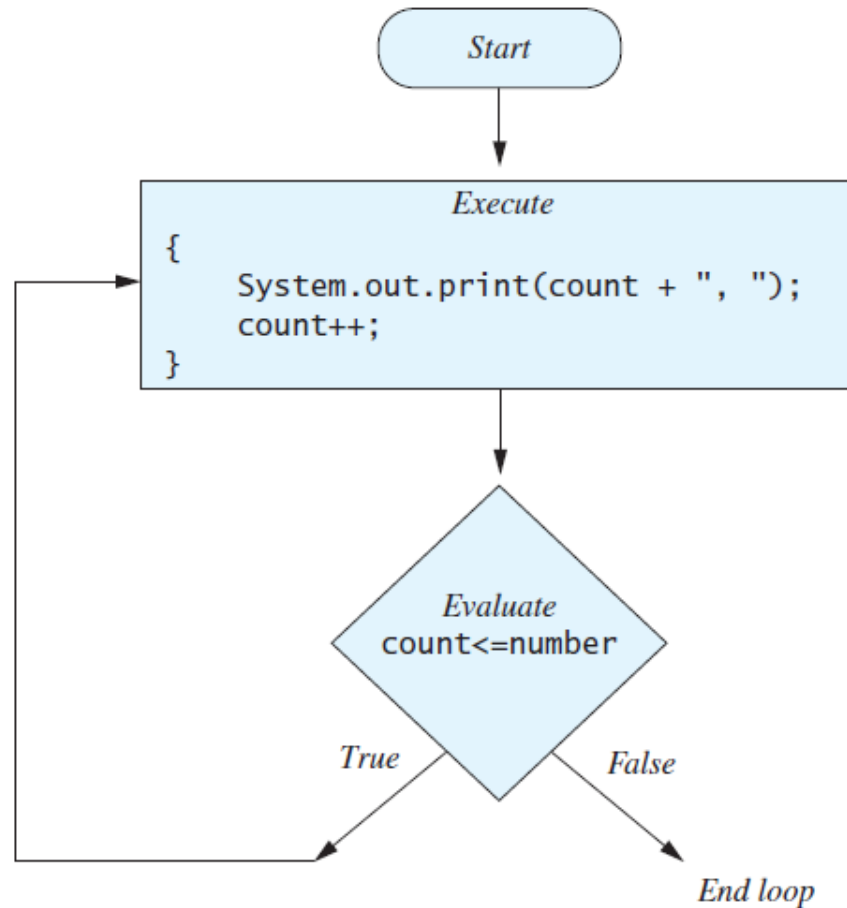
### Sample Screen Output 3

```
Enter a number:  
0  
1, ←  
Buckle my shoe.
```

*The loop body always  
executes at least once.*

**FIGURE 4.3** The Action of the `do-while` Loop in Listing 4.2

```
do
{
  System.out.print(count + ", ");
  count++;
} while (count <= number);
```

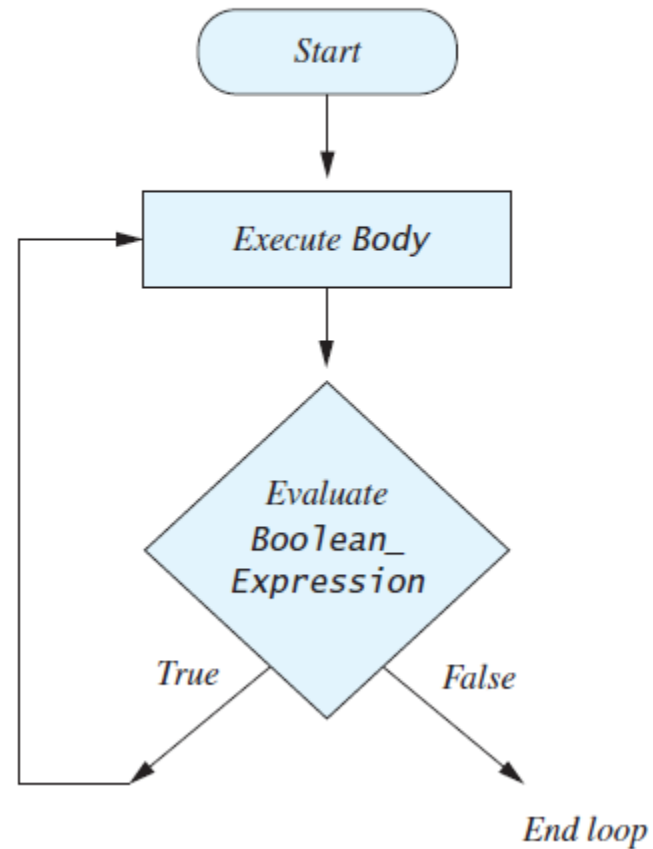




**FIGURE 4.4** The Semantics of the do-while Statement

---

```
do  
    Body  
while (Boolean_Expression)
```



### LISTING 4.3 Roach Population Program (part 1 of 2)

---

```
import java.util.Scanner;
/**
 * Program to calculate how long it will take a population of
 * roaches to completely fill a house from floor to ceiling.
 */
public class BugTrouble
{
    public static final double GROWTH_RATE = 0.95;    //95% per week
    public static final double ONE_BUG_VOLUME = 0.002; //cubic feet

    public static void main(String[] args)
    {
        System.out.println("Enter the total volume of your house");
        System.out.print("in cubic feet: ");
        Scanner keyboard = new Scanner(System.in);
        double houseVolume = keyboard.nextDouble();

        System.out.println("Enter the estimated number of");
        System.out.print("roaches in your house: ");
        int startPopulation = keyboard.nextInt();
        int countWeeks = 0;
        double population = startPopulation;
        double totalBugVolume = population * ONE_BUG_VOLUME;
        double newBugs, newBugVolume;

        while (totalBugVolume < houseVolume)
        {
            newBugs = population * GROWTH_RATE;
            newBugVolume = newBugs * ONE_BUG_VOLUME;
            population = population + newBugs;
            totalBugVolume = totalBugVolume + newBugVolume;
            countWeeks++;
        }
    }
}
```

*(int) is a  
type cast as  
discussed in  
Chapter 2.*

```
        System.out.println("Starting with a roach population of " +
                           startPopulation);
        System.out.println("and a house with a volume of " + houseVolume +
                           " cubic feet,");
        System.out.println("after " + countWeeks + " weeks,");
        System.out.println("the house will be filled with " +
                           (int)population + " roaches.");
        System.out.println("They will fill a volume of " +
                           (int)totalBugVolume + " cubic feet.");
        System.out.println("Better call Debugging Experts Inc.");
    }
}
```

---

### **Sample Screen Output**

```
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.
```

## LISTING 4.4 Nested Loops (part 1 of 2)

---

```
import java.util.Scanner;
/**
Computes the average of a list of (nonnegative) exam scores.
Repeats computation for more exams until the user says to stop.
*/
public class ExamAverager
{
    public static void main(String[] args)
    {
        System.out.println("This program computes the average of");
        System.out.println("a list of (nonnegative) exam scores.");
        double sum;
        int numberOfStudents;
        double next;
        String answer;
        Scanner keyboard = new Scanner(System.in);
```

```

do
{
    System.out.println();
    System.out.println("Enter all the scores to be averaged.");
    System.out.println("Enter a negative number after");
    System.out.println("you have entered all the scores.");
    sum = 0;
    numberOfStudents = 0;
    next = keyboard.nextDouble();
    while (next >= 0)
    {
        sum = sum + next;
        numberOfStudents++;
        next = keyboard.nextDouble();
    }
    if (numberOfStudents > 0)
        System.out.println("The average is " +
            (sum / numberOfStudents));
    else
        System.out.println("No scores to average.");

    System.out.println("Want to average another exam?");
    System.out.println("Enter yes or no.");
    answer = keyboard.next();
} while (answer.equalsIgnoreCase("yes"));
}
}

```

### Sample Screen Output

```
This program computes the average of  
a list of (nonnegative) exam scores.
```

```
Enter all the scores to be averaged.
```

```
Enter a negative number after  
you have entered all the scores.
```

```
100
```

```
90
```

```
100
```

```
90
```

```
-1
```

```
The average is 95.0
```

```
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.
```

```
no
```

## LISTING 4.5 An Example of a for Statement

---

```
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!");
    }
}
```

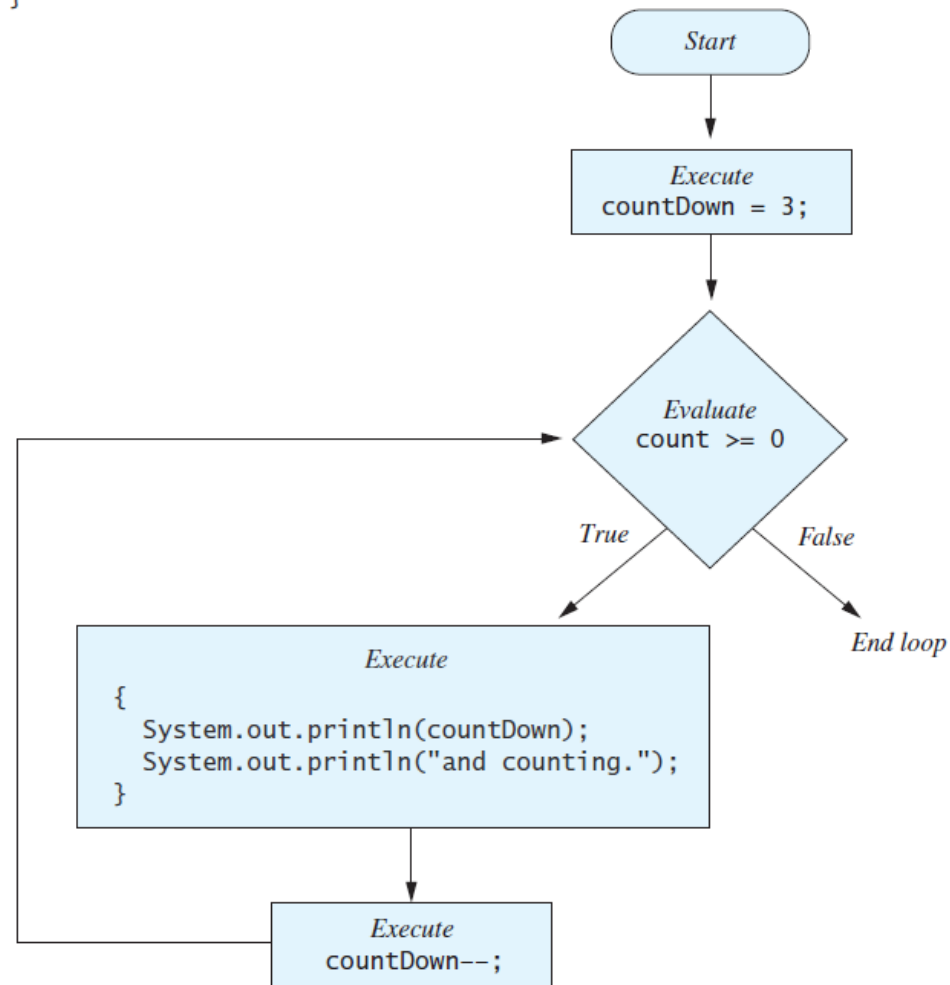
---

### Screen Output

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

**FIGURE 4.5** The Action of the `for` Loop in Listing 4.5

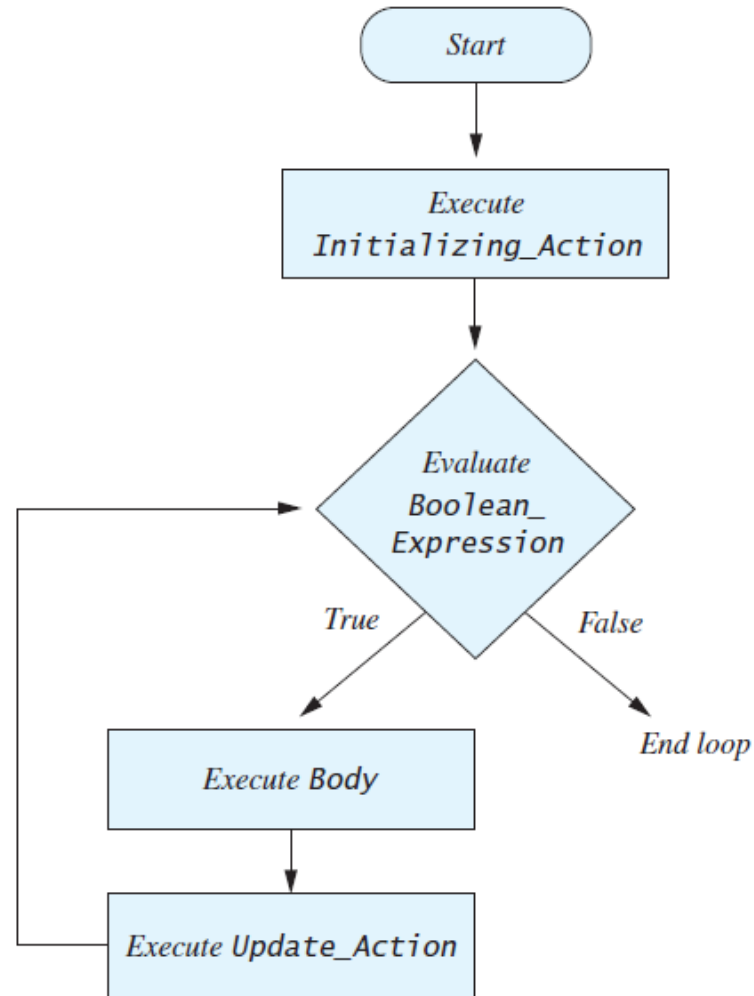
```
for (countDown = 3; countDown >= 0; countDown--)  
{  
    System.out.println(countDown);  
    System.out.println("and counting.");  
}
```





**FIGURE 4.6** The Semantics of the `for` Statement

`for` (*Initializing\_Action*; *Boolean\_Expression*; *Update\_Action*)  
*Body*



## LISTING 4.6 Using a Boolean Variable to End a Loop

---

```
import java.util.Scanner;
/**
 * Illustrates the use of a boolean variable to end loop iteration.
 */
public class BooleanDemo
{
    public static void main(String[] args)
    {
        System.out.println("Enter nonnegative numbers.");
        System.out.println("Place a negative number at the end");
        System.out.println("to serve as an end marker.");
        int sum = 0;
        boolean areMore = true;
        Scanner keyboard = new Scanner(System.in);
        while (areMore)
        {
            int next = keyboard.nextInt();
            if (next < 0)
                areMore = false;
            else
                sum = sum + next;
        }
        System.out.println("The sum of the numbers is " + sum);
    }
}
```

---

### Sample Screen Output

```
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
```

## LISTING 4.7 Spending Spree Program (part 1 of 2)

---

```
import java.util.Scanner;
public class SpendingSpree
{
    public static final int SPENDING_MONEY = 100;
    public static final int MAX_ITEMS = 3;
    public static void main(String[] args)
    {
        Scanner keyboard = new Scanner(System.in);
        boolean haveMoney = true;
        int leftToSpend = SPENDING_MONEY;
        int totalSpent = 0;
        int itemNumber = 1;
        while (haveMoney && (itemNumber <= MAX_ITEMS))
        {
            System.out.println("You may buy up to " +
                (MAX_ITEMS - itemNumber + 1) +
                " items");
            System.out.println("costing no more than $" +
                leftToSpend + ".");
            System.out.print("Enter cost of item #" +
                itemNumber + ": $");
            int itemCost = keyboard.nextInt();
            if (itemCost <= leftToSpend)
            {
                System.out.println("You may buy this item. ");
                totalSpent = totalSpent + itemCost;
                System.out.println("You spent $" + totalSpent +
                    " so far.");
                leftToSpend = SPENDING_MONEY - totalSpent;
                if (leftToSpend > 0)
                    itemNumber++;
                else
                {
                    System.out.println("You are out of money.");
                    haveMoney = false;
                }
            }
        }
    }
}
```

```
        else
            System.out.println("You cannot buy that item.");
        }
        System.out.println("You spent $" + totalSpent +
            ", and are done shopping.");
    }
}
```

---

### **Sample Screen Output**

```
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.
```


---

## LISTING 4.8 Ending a Loop with a break Statement

---

```
while (itemNumber <= MAX_ITEMS)
{
    . . .
    if (itemCost <= leftToSpend)
    {
        . . .
        if (leftToSpend > 0)
            itemNumber++;
        else
        {
            System.out.println("You are out of money.");
            break;
        }
    }
    . . .
}

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



## LISTING 4.9 An Applet That Uses Looping and Branching *(part 1 of 4)*

---

```
import javax.swing.JApplet;
import java.awt.Graphics;
import java.awt.Color;

public class MultipleFaces extends JApplet
{
    public static final int FACE_DIAMETER = 50;
    public static final int X_FACE0 = 10;
    public static final int Y_FACE0 = 5;

    public static final int EYE_WIDTH = 5;
    public static final int EYE_HEIGHT = 10;
    public static final int X_RIGHT_EYE0 = 20;
    public static final int Y_RIGHT_EYE0 = 15;
    public static final int X_LEFT_EYE0 = 45;
    public static final int Y_LEFT_EYE0 = Y_RIGHT_EYE0;

    public static final int NOSE_DIAMETER = 5;
    public static final int X_NOSE0 = 32;
    public static final int Y_NOSE0 = 25;

    public static final int MOUTH_WIDTH = 30;
    public static final int MOUTH_HEIGHT0 = 0;
    public static final int X_MOUTH0 = 20;
    public static final int Y_MOUTH0 = 35;
    public static final int MOUTH_START_ANGLE = 180;
    public static final int MOUTH_EXTENT_ANGLE = 180;
```

```

public void paint(Graphics canvas)
{
    int i, xOffset, yOffset; //Want i to exist after the loop ends
    for (i = 0; i <= 4; i++)
    { //Draw one face:
        xOffset = 50 * i;
        yOffset = 30 * i;

        //Draw face circle:
        if (i % 2 == 0) //if i is even
        { //Make face light gray
            canvas.setColor(Color.LIGHT_GRAY);
            canvas.fillOval(X_FACE0 + xOffset, Y_FACE0 + 30 * i,
                           FACE_DIAMETER, FACE_DIAMETER);
        }
        canvas.setColor(Color.BLACK);
        canvas.drawOval(X_FACE0 + xOffset, Y_FACE0 + yOffset,
                       FACE_DIAMETER, FACE_DIAMETER);
    }
}

```

```

//Draw eyes:
canvas.setColor(Color.BLUE);
canvas.fillOval(X_RIGHT_EYE0 + xOffset, Y_RIGHT_EYE0 +
               yOffset, EYE_WIDTH, EYE_HEIGHT);
canvas.fillOval(X_LEFT_EYE0 + xOffset, Y_LEFT_EYE0 +
               yOffset, EYE_WIDTH, EYE_HEIGHT);

//Draw nose:
canvas.setColor(Color.BLACK);
canvas.fillOval(X_NOSE0 + xOffset, Y_NOSE0 + yOffset,
               NOSE_DIAMETER, NOSE_DIAMETER);

//Draw mouth:
canvas.setColor(Color.RED);
canvas.drawArc(X_MOUTH0 + xOffset, Y_MOUTH0 + yOffset,
              MOUTH_WIDTH, MOUTH_HEIGHT0 + 3 * i,
              MOUTH_START_ANGLE, MOUTH_EXTENT_ANGLE);
}
//i is 5 when the previous loop ends ←
xOffset = 50 * i;
yOffset = 30 * i;

//Draw kissing face:
//Draw face outline:
canvas.setColor(Color.BLACK);
canvas.drawOval(X_FACE0 + xOffset, Y_FACE0 + yOffset,
               FACE_DIAMETER, FACE_DIAMETER);

//Draw eyes:
canvas.setColor(Color.BLUE);
canvas.fillOval(X_RIGHT_EYE0 + xOffset, Y_RIGHT_EYE0 + yOffset,
               EYE_WIDTH, EYE_HEIGHT);
canvas.fillOval(X_LEFT_EYE0 + xOffset, Y_LEFT_EYE0 + yOffset,
               EYE_WIDTH, EYE_HEIGHT);

//Draw nose:
canvas.setColor(Color.BLACK);
canvas.fillOval(X_NOSE0 + xOffset, Y_NOSE0 + yOffset,
               NOSE_DIAMETER, NOSE_DIAMETER);

//Draw mouth in shape of a kiss:
canvas.setColor(Color.RED);
canvas.fillOval(X_MOUTH0 + xOffset + 10, Y_MOUTH0 + yOffset,
               MOUTH_WIDTH - 20, MOUTH_WIDTH - 20);

```

After the last iteration of the loop body, the value of *i* is incremented one last time to become 5.



```

//Add text:
canvas.drawString("Kiss, Kiss.",
    X_FACE0 + xOffset + FACE_DIAMETER, Y_FACE0 + yOffset);

//Draw blushing face:
i++;
xOffset = 50 * i;
yOffset = 30 * i;

//Draw face circle:
canvas.setColor(Color.GRAY);
canvas.fillOval(X_FACE0 + xOffset, Y_FACE0 + yOffset,
    FACE_DIAMETER, FACE_DIAMETER);
canvas.setColor(Color.BLACK);
canvas.drawOval(X_FACE0 + xOffset, Y_FACE0 + yOffset,
    FACE_DIAMETER, FACE_DIAMETER);

//Draw eyes:
canvas.setColor(Color.BLACK);
canvas.fillOval(X_RIGHT_EYE0 + xOffset, Y_RIGHT_EYE0 +
    yOffset, EYE_WIDTH, EYE_HEIGHT);
canvas.fillOval(X_LEFT_EYE0 + xOffset, Y_LEFT_EYE0 + yOffset,
    EYE_WIDTH, EYE_HEIGHT);

//Draw nose:
canvas.setColor(Color.BLACK);
canvas.fillOval(X_NOSE0 + xOffset, Y_NOSE0 + yOffset,
    NOSE_DIAMETER, NOSE_DIAMETER);

//Draw mouth:
canvas.setColor(Color.BLACK);
canvas.drawArc(X_MOUTH0 + xOffset, Y_MOUTH0 + yOffset,
    MOUTH_WIDTH, MOUTH_HEIGHT0 + 3 * (i - 2),
    MOUTH_START_ANGLE, MOUTH_EXTENT_ANGLE);

//Add text:
canvas.drawString("Tee Hee.",
    X_FACE0 + xOffset + FACE_DIAMETER, Y_FACE0 + yOffset);
}
}

```

## Applet Output

