For-loops

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

What is a for-loop?

- A **loop** $\cdots$ a program structure where a block of the code is executed repeatedly
- The repetition continues as long as a specific condition holds

Three Types of Loops in Java

There are three types of loops in Java: for, while, and do-while

For-loops: a for-loop is a structure that encompasses its loop body with three components:

- *Initialization* (a single statement)
- *Continuation condition* (a condition generating statement)
- *Update* (a single statement)

General For-loop Format

The general format for a for-loop is:

```
for (INITIALIZATION; CONTINUATION CONDITION; UPDATE) {
  loop-body
}
```

General For-loop Format

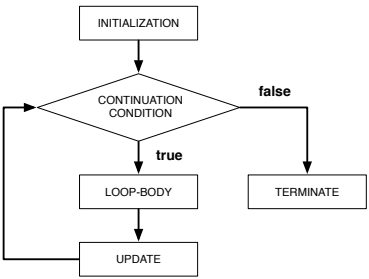
The general format for a for-loop is:

```
for (INITIALIZATION; CONTINUATION CONDITION; UPDATE) {
  loop-body
}
```

This is interpreted as:

- Execute INITIALIZATION
- As long as the CONTINUATION CONDITION holds
  - Execute the loop-body
  - Execute UPDATE

Flowchart for a For-loop



An Example

```
1  public class ForExample {
2    public static void main( String[] args ) {
3      int count;
4      for ( count = 1; count <= 8; count ++ ) {
5        System.out.println( "The value of count is " + count );
6      }
7    }
8  }
```

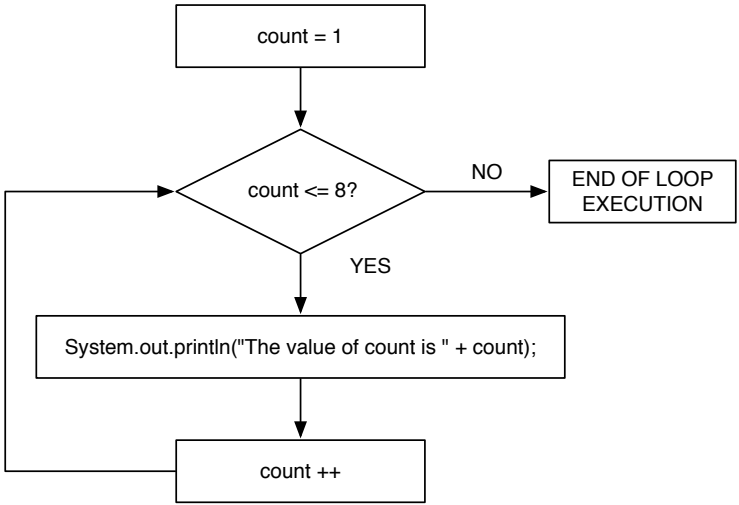
INITIALIZATION: set count to 1
CONTINUATION: as long as count is less than or equal to 8
UPDATE: add 1 to count

An Example

```
1  public class ForExample {
2    public static void main( String[] args ) {
3      int count;
4      for ( count = 1; count <= 8; count ++ ) {
5        System.out.println( "The value of count is " + count );
6      }
7    }
8  }
```

Loop body: print the `String` literal `"The value of count is "` followed by the value of `count`

Flowchart


count = 1
count <= 8?
NO
END OF LOOP EXECUTION
YES
System.out.println("The value of count is " + count);
count ++

Inserting Type Declaration into the Loop Definition

You may add the variable type to the iterator variable initialization in the for loop, for example:

```
for (int count = 1; count <= 8; count ++) {
  System.out.println("The value of count is " + count);
}
```

Inserting Type Declaration into the Loop Definition

You may add the variable type to the iterator variable initialization in the for loop, for example:

```
for (int count = 1; count <= 8; count ++) {
  System.out.println("The value of count is " + count);
}
```

Note that the scope of `count` is the entire for loop

Loop Variants

```
for (count = 10; count >= 0; count --)
```
`count` goes 10, 9, 8, . . ., 0

```
for (count = 10; count >= 0; count -= 2)
```
`count` goes 10, 8, 6, . . ., 0

```
for (count = 0; count < 10; count += 2)
```
`count` goes 0, 2, 4, 6, 8

Counting Down

```
1  public class CountDown {
2    public static void main( String[] args ) {
3      for ( int value = 100; value >= 1; value -- ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

The for-loop iterates the variable `value` from the value 100 downward, decreasing the value by 1 each time; quits as soon as the value becomes less than 1

Counting Down

```
1  public class CountDown {
2    public static void main( String[] args ) {
3      for ( int value = 100; value >= 1; value -- ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

The loop body simply prints out the value of the variable with two dots preceding

Counting Down

```
1  public class CountDown {
2    public static void main( String[] args ) {
3      for ( int value = 100; value >= 1; value -- ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

Upon quitting the loop, print a message

Accelerated Counting Down

```
1  public class CountDownFast {
2    public static void main( String[] args ) {
3      for ( int value = 1000000; value >= 1; value /= 2 ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

The for-loop iterates the variable `value` from one million, dividing it by two each time; the loop quits as soon as `value` becomes less than 1

Accelerated Counting Down

```
1  public class CountDownFast {
2    public static void main( String[] args ) {
3      for ( int value = 1000000; value >= 1; value /= 2 ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

The loop body simply prints out the value of the variable with two dots preceding

Accelerated Counting Down

```
1  public class CountDownFast {
2    public static void main( String[] args ) {
3      for ( int value = 1000000; value >= 1; value /= 2 ) {
4        System.out.println( ".." + value );
5      }
6      System.out.println( "BOOOOOOOOOOOOOOOOOOOOOOOM!" );
7    }
8  }
```

Upon quitting the loop, print a message

Computing Squares

Compute the squares i^2 for $i = 1, \ldots, 20$

Computing Squares

Compute the squares i^2 for $i = 1, \ldots, 20$

```
1  public class Squares {
2    public static void main( String[] args ) {
3      int square;
4      for ( int i = 1; i <= 20; i ++ ) {
5        square = i * i;
6        System.out.print( "The square of " + i );
7        System.out.println( " is " + square );
8      }
9    }
10 }
```

For loop for generating $1, \ldots, 20$

Computing Squares

Compute the squares i^2 for $i = 1, \ldots, 20$

```
1  public class Squares {
2    public static void main( String[] args ) {
3      int square;
4      for ( int i = 1; i <= 20; i ++ ) {
5        square = i * i;
6        System.out.print( "The square of " + i );
7        System.out.println( " is " + square );
8      }
9    }
10 }
```

Compute the square in the variable `square`

Computing Squares, Alternative

Compute the squares i^2 for $i = 1, \ldots, 20$
Use the fact i^2 is equal to the sum of the first i odd numbers

Computing Squares, Alternative

Compute the squares i^2 for $i = 1, \ldots, 20$
Use the fact i^2 is equal to the sum of the first i odd numbers

```
1  public class SquaresAlt {
2    public static void main( String[] args ) {
3      int square = 1;
4      for ( int i = 1; i <= 20; i ++ ) {
5        square += 2 * i + 1;
6        System.out.print( "The square of " + i );
7        System.out.println( " is " + square );
8      }
9    }
10 }
```

The initial value of `square` is 0

Computing Squares, Alternative

Compute the squares i^2 for $i = 1, \ldots, 20$
Use the fact i^2 is equal to the sum of the first i odd numbers

```
1  public class SquaresAlt {
2    public static void main( String[] args ) {
3      int square = 1;
4      for ( int i = 1; i <= 20; i ++ ) {
5        square += 2 * i + 1;
6        System.out.print( "The square of " + i );
7        System.out.println( " is " + square );
8      }
9    }
10 }
```

For loop for generating $1, \ldots, 20$

Computing Squares, Alternative

Compute the squares i^2 for $i = 1, \ldots, 20$
Use the fact i^2 is equal to the sum of the first i odd numbers

```
1  public class SquaresAlt {
2    public static void main( String[] args ) {
3      int square = 1;
4      for ( int i = 1; i <= 20; i ++ ) {
5        square += 2 * i + 1;
6        System.out.print( "The square of " + i );
7        System.out.println( " is " + square );
8      }
9    }
10 }
```

Compute the square by adding $2 * i + 1$

Summing from 1 to 100 Every Third Number

Compute the sum of numbers 1, 4, 7, . . ., 100

Summing from 1 to 100 Every Third Number

Compute the sum of numbers 1, 4, 7, . . ., 100

```
1  // calculate the sum 1 + 4 + ... + 100
2  public class SumTo100EveryThird {
3      public static void main( String[] args ) {
4          //-- initialize the sum
5          long sum = 0;
6          //-- iterate the value of j from 1 to 100
7          for ( int count = 1; count <= 100; count += 3 ) {
8              sum += count;
9          }
10         System.out.println( "1 + 4 + ... + 100 = " + sum );
11     }
12 }
```

Set the total value `sum` to 0

Summing from 1 to 100 Every Third Number

Compute the sum of numbers 1, 4, 7, . . ., 100

```
1  // calculate the sum 1 + 4 + ... + 100
2  public class SumTo100EveryThird {
3      public static void main( String[] args ) {
4          //-- initialize the sum
5          long sum = 0;
6          //-- iterate the value of j from 1 to 100
7          for ( int count = 1; count <= 100; count += 3 ) {
8              sum += count;
9          }
10         System.out.println( "1 + 4 + ... + 100 = " + sum );
11     }
12 }
```

Iterate the value of `count` from 1 through 100, adding 3 each time

Summing from 1 to 100 Every Third Number

Compute the sum of numbers 1, 4, 7, . . ., 100

```
1  // calculate the sum 1 + 4 + ... + 100
2  public class SumTo100EveryThird {
3      public static void main( String[] args ) {
4          //-- initialize the sum
5          long sum = 0;
6          //-- iterate the value of j from 1 to 100
7          for ( int count = 1; count <= 100; count += 3 ) {
8              sum += count;
9          }
10         System.out.println( "1 + 4 + ... + 100 = " + sum );
11     }
12 }
```

Add the value of `count` to `sum`

Summing from 1 to 100 Every Third Number

Compute the sum of numbers 1, 4, 7, . . ., 100

```
1  // calculate the sum 1 + 4 + ... + 100
2  public class SumTo100EveryThird {
3      public static void main( String[] args ) {
4          //-- initialize the sum
5          long sum = 0;
6          //-- iterate the value of j from 1 to 100
7          for ( int count = 1; count <= 100; count += 3 ) {
8              sum += count;
9          }
10         System.out.println( "1 + 4 + ... + 100 = " + sum );
11     }
12 }
```

Print the outcome

The `String` literal `"1 + 4 + ... + 100 = "` followed by the total; the two parts are concatenated inside the println statement

Table of Contents

Loops May Exist in a Loop Body

A for-loop may appear in the loop body of a for-loop

Loops May Exist in a Loop Body

A for-loop may appear in the loop body of a for-loop

```
int count1, count2;
for (count1 = X; ...; ...)  {
  ...
  for (count2 = Y; ...  ; ...)  {
      ...
  }
  ...
}
```

Loops May Exist in a Loop Body

A for-loop may appear in the loop body of a for-loop

```
int count1, count2;
for (count1 = X; ...; ...)  {
  ...
  for (count2 = Y; ...  ; ...)  {
      ...
  }
  ...
}
```

Naturally we want `count1` and `count2` to be different

The use of `count1` in place of `count2` may result in a serious logic error

Box

Receive height and width from user and print a box of the dimensions

```
1  public class DarkBox {
2    //-- main method
3    public static void main( String[] args ) {
4      int height, width;
5      // set height and width
6      height = 10;
7      width = 50;
8      // exterior loop
9      for ( int posV = 1; posV <= height; posV ++ ) {
10       // interior loop
11       for ( int posH = 1; posH <= width; posH ++ ) {
12         System.out.print( "#" );
13       }
14       System.out.println();
15     }
16   }
17 }
```

Variable declarations and value assignments

Box

Receive height and width from user and print a box of the dimensions

```
1  public class DarkBox {
2    //-- main method
3    public static void main( String[] args ) {
4      int height, width;
5      // set height and width
6      height = 10;
7      width = 50;
8      // exterior loop
9      for ( int posV = 1; posV <= height; posV ++ ) {
10       // interior loop
11       for ( int posH = 1; posH <= width; posH ++ ) {
12         System.out.print( "#" );
13       }
14       System.out.println();
15     }
16   }
17 }
```

The external loop iterates `posV` from 1 to `height`

Box

Receive height and width from user and print a box of the dimensions

```
1  public class DarkBox {
2    //-- main method
3    public static void main( String[] args ) {
4      int height, width;
5      // set height and width
6      height = 10;
7      width = 50;
8      // exterior loop
9      for ( int posV = 1; posV <= height; posV ++ ) {
10       // interior loop
11       for ( int posH = 1; posH <= width; posH ++ ) {
12         System.out.print( "#" );
13       }
14       System.out.println();
15     }
16   }
17 }
```

The internal loop iterates `posH` from 1 to `width`

Box

Receive height and width from user and print a box of the dimensions

```
1  public class DarkBox {
2    //-- main method
3    public static void main( String[] args ) {
4      int height, width;
5      // set height and width
6      height = 10;
7      width = 50;
8      // exterior loop
9      for ( int posV = 1; posV <= height; posV ++ ) {
10       // interior loop
11       for ( int posH = 1; posH <= width; posH ++ ) {
12         System.out.print( "#" );
13       }
14       System.out.println();
15     }
16   }
17 }
```

Inside the internal loop print one `#` for each value of `posH`
This generates `width` many `#`'s successively

Box

Receive height and width from user and print a box of the dimensions

```
1  public class DarkBox {
2    //-- main method
3    public static void main( String[] args ) {
4      int height, width;
5      // set height and width
6      height = 10;
7      width = 50;
8      // exterior loop
9      for ( int posV = 1; posV <= height; posV ++ ) {
10       // interior loop
11       for ( int posH = 1; posH <= width; posH ++ ) {
12         System.out.print( "#" );
13       }
14       System.out.println();
15     }
16   }
17 }
```

After concluding the inner loop go to the new line

Framed Box

Similar to the previous one, but surround the box with a line

- Attach a '|' before and after each line
- Attach "+———-+" at the top and the bottom, where the number of '-' is equal to the width.

Example

With dimension = 10*10

```
% java FramedBox
+----------+
|##########|
|##########|
|##########|
|##########|
|##########|
|##########|
|##########|
|##########|
|##########|
+----------+
```

Framed Box

Previously we used the following double for-loop

```
1  for ( int posV = 1; posV <= height; posV ++ ) {
2    for ( int posH = 1; posH <= width; posH ++ ) {
3      System.out.print( "#" );
4    }
5    System.out.println();
6  }
```

We change this to

```
1  for ( int posV = 1; posV <= height; posV ++ ) {
2    System.out.print( "|" );
3    for ( int posH = 1; posH <= width; posH ++ ) {
4      System.out.print( "#" );
5    }
6    System.out.println( "|" );
7  }
```

This adds a | before and after each line of #'s

Framed Box

We also add the following before and after:

```
1 System.out.print( "+" );
2 for ( int posH = 1; posH <= width; posH ++ ) {
3   System.out.print( "-" );
4 }
5 System.out.println( "+" );
```

This produces a `+`, `width` many `-`'s, and a `+` and goes to the next line

Framed Box

Similar to the previous one, but surround the box with a line

Framed Box

Similar to the previous one, but surround the box with a line

```
1  public class FramedBox {
2    public static void main( String[] args ) {
3      int height = 10, width = 50;
4      //--- the top line ---//
5      System.out.print( "+" );
6      for ( int posH = 1; posH <= width; posH ++ ) {
7        System.out.print( "-" );
8      }
9      System.out.println( "+" );
10     //--- the middle lines ---//
11     for ( int posV = 1; posV <= height; posV ++ ) {
12       System.out.print( "|" );
13       for ( int posH = 1; posH <= width; posH ++ ) {
14         System.out.print( "#" );
15       }
16       System.out.println( "|" );
17     }
18     //--- the bottom line---//
19     System.out.print( "+" );
20     for ( int posH = 1; posH <= width; posH ++ ) {
21       System.out.print( "-" );
22     }
23     System.out.println( "+" );
24   }
25 }
```

Declaring the dimension variables and assigning values to them

Framed Box

Similar to the previous one, but surround the box with a line

```
1  public class FramedBox {
2    public static void main( String[] args ) {
3      int height = 10, width = 50;
4      //--- the top line ---//
5      System.out.print( "+" );
6      for ( int posH = 1; posH <= width; posH ++ ) {
7        System.out.print( "-" );
8      }
9      System.out.println( "+" );
10     //--- the middle lines ---//
11     for ( int posV = 1; posV <= height; posV ++ ) {
12       System.out.print( "|" );
13       for ( int posH = 1; posH <= width; posH ++ ) {
14         System.out.print( "#" );
15       }
16       System.out.println( "|" );
17     }
18     //--- the bottom line---//
19     System.out.print( "+" );
20     for ( int posH = 1; posH <= width; posH ++ ) {
21       System.out.print( "-" );
22     }
23     System.out.println( "+" );
24   }
25 }
```

The top part: a '+', dashes, and then a '+'

Framed Box

Similar to the previous one, but surround the box with a line

```
1  public class FramedBox {
2    public static void main( String[] args ) {
3      int height = 10, width = 50;
4      //--- the top line ---//
5      System.out.print( "+" );
6      for ( int posH = 1; posH <= width; posH ++ ) {
7        System.out.print( "-" );
8      }
9      System.out.println( "+" );
10     //--- the middle lines ---//
11     for ( int posV = 1; posV <= height; posV ++ ) {
12       System.out.print( "|" );
13       for ( int posH = 1; posH <= width; posH ++ ) {
14         System.out.print( "#" );
15       }
16       System.out.println( "|" );
17     }
18     //--- the bottom line---//
19     System.out.print( "+" );
20     for ( int posH = 1; posH <= width; posH ++ ) {
21       System.out.print( "-" );
22     }
23     System.out.println( "+" );
24   }
25 }
```

The middle part

Framed Box

Similar to the previous one, but surround the box with a line

```
1  public class FramedBox {
2    public static void main( String[] args ) {
3      int height = 10, width = 50;
4      //--- the top line ---//
5      System.out.print( "+" );
6      for ( int posH = 1; posH <= width; posH ++ ) {
7        System.out.print( "-" );
8      }
9      System.out.println( "+" );
10     //--- the middle lines ---//
11     for ( int posV = 1; posV <= height; posV ++ ) {
12       System.out.print( "|" );
13       for ( int posH = 1; posH <= width; posH ++ ) {
14         System.out.print( "#" );
15       }
16       System.out.println( "|" );
17     }
18     //--- the bottom line---//
19     System.out.print( "+" );
20     for ( int posH = 1; posH <= width; posH ++ ) {
21       System.out.print( "-" );
22     }
23     System.out.println( "+" );
24   }
25 }
```

The bottom part, the same as the top part

Inverted Triangle

Print the right-angle triangle of a fixed height

- The triangular area will be represented with a $\star$
- The first line has the height-many $\star$'s
- The second line the (height-1) many $\star$'s
- ...
- The last line has one $\star$

All the lines will print left-flushed

Example

```
% java TriangleFlipped
******************
*****************
****************
***************
**************
*************
************
***********
**********
*********
********
*******
******
*****
****
***
**
*
```

Flipped Triangle

Print an inverted triangle

```
1  //-- print a triangle
2  public class TriangleFlipped {
3    public static void main( String[] args ) {
4      int height = 18;
5      for ( int posV = height; posV >= 1; posV -- ) {
6        for ( int posH = 1; posH <= posV; posH ++ ) {
7          System.out.print( "*" );
8        }
9        System.out.println();
10     }
11   }
12 }
```

Height specification

Flipped Triangle

Print an inverted triangle

```
1  //-- print a triangle
2  public class TriangleFlipped {
3    public static void main( String[] args ) {
4      int height = 18;
5      for ( int posV = height; posV >= 1; posV -- ) {
6        for ( int posH = 1; posH <= posV; posH ++ ) {
7          System.out.print( "*" );
8        }
9        System.out.println();
10     }
11   }
12 }
```

External loop is executed with the variable `posV` iterated from `height` down to 1

Flipped Triangle

Print an inverted triangle

```
1  //-- print a triangle
2  public class TriangleFlipped {
3    public static void main( String[] args ) {
4      int height = 18;
5      for ( int posV = height; posV >= 1; posV -- ) {
6        for ( int posH = 1; posH <= posV; posH ++ ) {
7          System.out.print( "*" );
8        }
9        System.out.println();
10     }
11   }
12 }
```

The internal loop produces `posV` many `*`'s
The `println` is for moving to the next line

Triple Loops

Consider the problem of printing in a single line '1' once, '2' twice, . . ., '9' nine times

The code can be

```
1 for ( int number = 1; number <= 9; number ++ ) {
2   for ( int count = 1; count <= number; count ++ ) {
3     System.out.print( number );
4   }
5 }
6 System.out.println();
```

What if we want to generate 9 lines, where the *i*-th line is truncated after output *i* as in

```
1 1
2 122
3 122333
4 1223334444
5 122333444455555
6 12233344455555666666
7 1223334444555556666667777777
8 122333444455555666666777777788888888
9 12233344445555566666677777778888888899999999
```

Solution

```
1  public class NumberSequences {
2    public static void main( String[] args ) {
3      for ( int line = 1; line <= 9; line ++ ) {
4        for ( int number = 1; number <= line; number ++ ) {
5          for ( int count = 1; count <= number; count ++ ) {
6            System.out.print( number );
7          }
8        }
9        System.out.println();
10     }
11   }
12 }
```

The line number coincides with the maximum number to print

Solution

```
1  public class NumberSequences {
2    public static void main( String[] args ) {
3      for ( int line = 1; line <= 9; line ++ ) {
4        for ( int number = 1; number <= line; number ++ ) {
5          for ( int count = 1; count <= number; count ++ ) {
6            System.out.print( number );
7          }
8        }
9        System.out.println();
10     }
11   }
12 }
```

Individial line; uses the previous double loop with `line` substituting 9

Table of Contents

Defining Constants

A constant in a program is a value with a name that is used throughout the program with no possibility of changing value during the execution of the program

To define a constant use

```
public static final <type> <name> = <value>;
```

at the same level as methods, where `final` means that the variable can be assigned a value only once

Usually they appear immediately after class declartion

Also, we use all capitals for constants

Framed Box with Size Constants

Use four parameters:

- `SCREEN_WIDTH` and `SCREEN_HEIGHT`: hypothetical width and height of the screen area
- `INTERIOR_WIDTH` and `INTERIOR_HEIGHT`: the previous values minus 2 - substract character space for the frame

```
1  public class FramedBoxConstant {
2    //---- dimensional information
3    public static final int SCREEN_WIDTH = 72;
4    public static final int SCREEN_HEIGHT = 20;
5    public static final int INTERIOR_WIDTH = SCREEN_WIDTH - 2;
6    public static final int INTERIOR_HEIGHT = SCREEN_HEIGHT - 2;
```

The constants have all-capital names
The second pair are defined based upon the first two

The Remainder of the Code

```
7     //---- the top and the bottom
8     public static void topOrBottom() {
9       System.out.print( "+" );
10      for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
11        System.out.print( "-" );
12      }
13      System.out.println( "+" );
14    }
15    //---- the middle part
16    public static void middle() {
17      for ( int posV = 1; posV <= INTERIOR_HEIGHT; posV ++ ) {
18        System.out.print( "|" );
19        for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
20          System.out.print( "#" );
21        }
22        System.out.println( "|" );
23      }
24    }
25    //---- main
26    public static void main( String[] args ) {
27      topOrBottom();
28      middle();
29      topOrBottom();
30    }
31  }
```

Method for the top line and for the bottom line

The Remainder of the Code

```
7     //---- the top and the bottom
8     public static void topOrBottom() {
9       System.out.print( "+" );
10      for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
11        System.out.print( "-" );
12      }
13      System.out.println( "+" );
14    }
15    //---- the middle part
16    public static void middle() {
17      for ( int posV = 1; posV <= INTERIOR_HEIGHT; posV ++ ) {
18        System.out.print( "|" );
19        for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
20          System.out.print( "#" );
21        }
22        System.out.println( "|" );
23      }
24    }
25    //---- main
26    public static void main( String[] args ) {
27      topOrBottom();
28      middle();
29      topOrBottom();
30    }
31  }
```

Method for the middle part

The Remainder of the Code

```
7     //---- the top and the bottom
8     public static void topOrBottom() {
9       System.out.print( "+" );
10      for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
11        System.out.print( "-" );
12      }
13      System.out.println( "+" );
14    }
15    //---- the middle part
16    public static void middle() {
17      for ( int posV = 1; posV <= INTERIOR_HEIGHT; posV ++ ) {
18        System.out.print( "|" );
19        for ( int posH = 1; posH <= INTERIOR_WIDTH; posH ++ ) {
20          System.out.print( "#" );
21        }
22        System.out.println( "|" );
23      }
24    }
25    //---- main
26    public static void main( String[] args ) {
27      topOrBottom();
28      middle();
29      topOrBottom();
30    }
31  }
```

Main uses the two methods to generate the shape

The End