



CMSC 131

Object-Oriented Programming I

For Statement, Nested Loops

**Dept of Computer Science
University of Maryland College Park**

This material is based on material provided by Ben Bederson, Bonnie Dorr,
Fawzi Emad, David Mount, Jan Plane

Overview

- ▶ For loops
- ▶ Nested Loops
- ▶ Expressions side effects
- ▶ Assignment operators

Types of loops

▶ Indefinite iteration

- Usually tests something that is coming from outside the loop structure (e.g. input)
- Needs to eventually change from true to false

▶ Counted iteration

- Something that is controlled inside the loop
- To start at some value and count up or down until some set ending point

for loop

- ▶ **for-loop** → The counter is set, the condition is tested before each body execution, the update is performed at the end of each iteration
for (initialization; condition; update) {
 ⟨body⟩
}
- ▶ Usually used for counted loops, but any of the parts can be left empty
- ▶ **Example:** ForExample.java

Infinite Loops

- ▶ Loops can run forever if condition never becomes false
- ▶ Be careful when programming loops!
 - Add statements for termination into loop body first
 - Often these statements are at end of body
 - e.g.

```
while (i <= 10) {  
    System.out.println(i);  
    i = i + 1;  
}
```

Nested Loops

- ▶ while, do-while are statement constructors (like if and if-else: they use blocks)
- ▶ Loops can thus be used inside other loops!
- ▶ **Example:** NestedWhile.java, NestedFor.java
- ▶ Let's build a trace table for NestedWhile.java

About Local Variables

- ▶ When you declare local variables they are only accessible (in scope) within the block they are declared in
- ▶ **Example:** ScopeError.java

Expressions

- ▶ Java “expressions” that yield values

e.g.

`x`

`x + 1 - y`

`x == y && z == 0`

`foo.equals ("cat")`

- ▶ Expressions have values of a specific type (int, boolean, etc.)
- ▶ Expressions can be assigned to variables, appear inside other expressions, etc.

Expressions and Side Effects

- ▶ Some expressions can also alter the values of variables
e.g. $x=1$
- ▶ $x=1$ is an expression?
 - Yes!
 - Value is result of evaluation right-hand side of $=$
 - It also alters the value of x
- ▶ Such alterations are called **side effects**

Are the Following Legal?

- ▶ `int x, y;`
`x = y = 1;`

Yes. Result assigns 1 to x and to y

- ▶ `int x = 0, y = 1;`
`boolean b = false;`
`if (b = (x <= y)){`
 `x = y;`
`}`

Yes. Result assigns true to b and 1 to x

Other Assignment Operators

- ▶ Example: **decrement** operations (Basically equivalent to $x = x - 1$)

--x "Pre-decrement"

Decrement x , returns the new value of x

x-- "Post-decrement"

Decrement x , returns the old value of x

"return x , then decrement it"

- ▶ General modification by constant
 - General form: **<var> <op with=> <constant>**
 - Examples

$x += 2$	equivalent to	$x = x + 2$
----------	---------------	-------------

$x -= 2$	equivalent to	$x = x - 2$
----------	---------------	-------------

$x *= 2$	equivalent to	$x = x * 2$
----------	---------------	-------------

$x /= 2$	equivalent to	$x = x / 2$
----------	---------------	-------------

Examples

- ▶ Let's try to draw shapes with asterisks
 - Horizontal line
 - Vertical line
 - Square
 - Triangle of asterisks