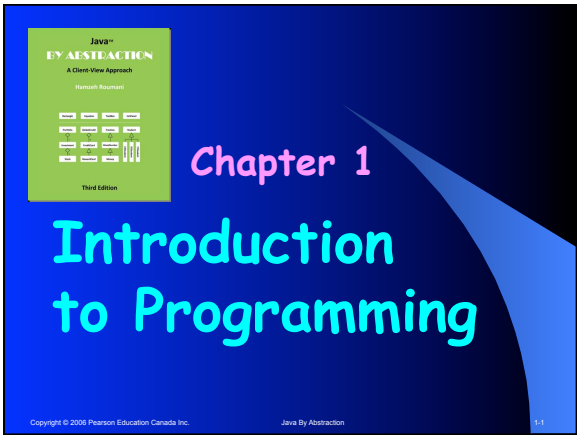


Chapter 1
Introduction to Programming

 Java By Abstraction 1-1

Outline

 Java By Abstraction 1-2

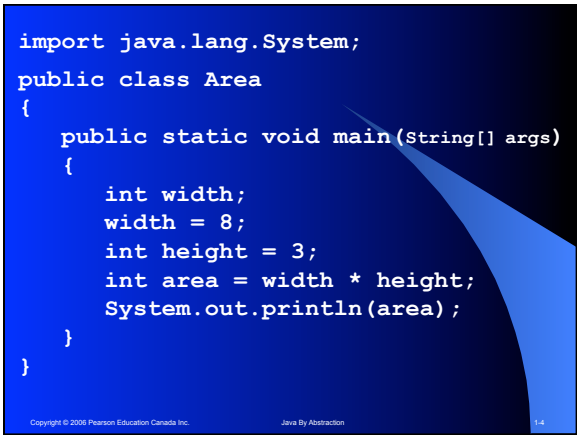
1.1.1 A Quick Tour

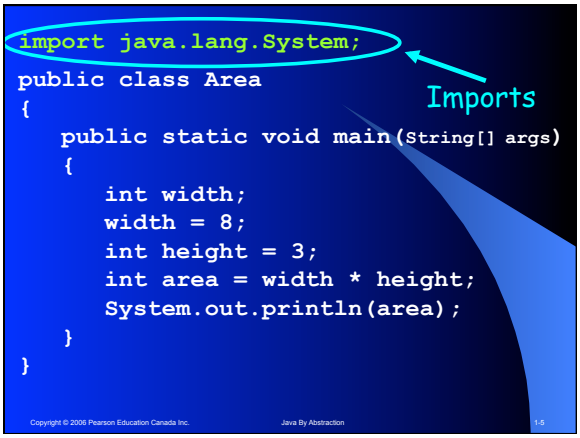
Let us take a look at a Java program. It does not matter now how the program was written; just become familiar with the terminology for describing program structures.

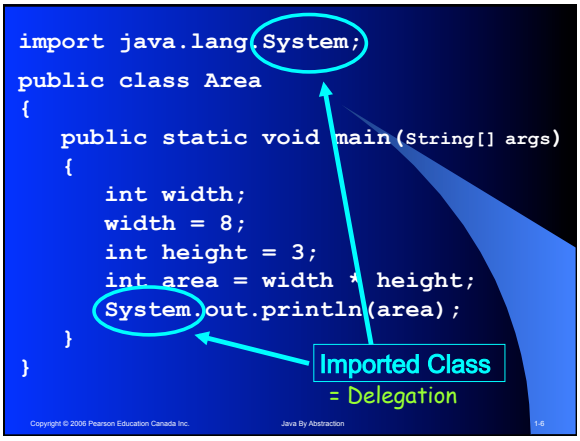
Note, in particular, the following four terms:

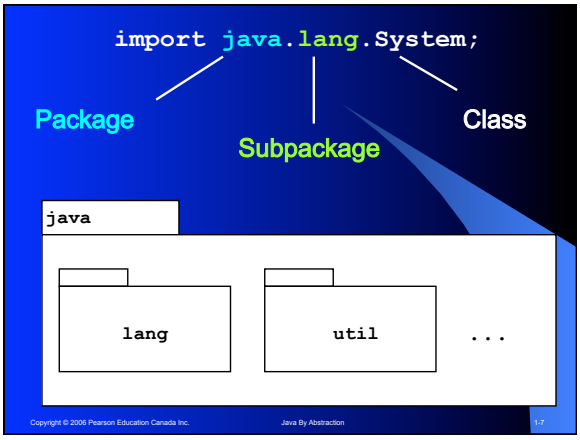
Imports, Class, Method, Style

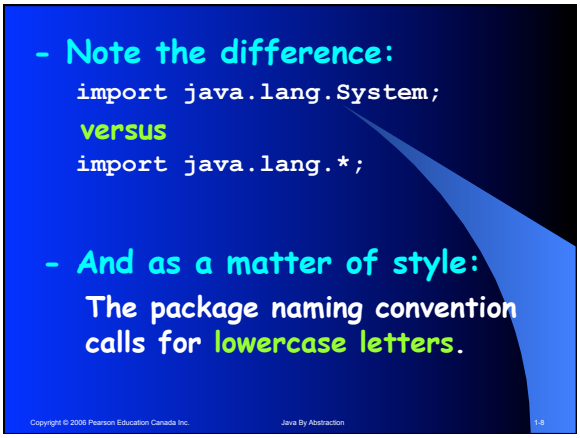
 Java By Abstraction 1-3

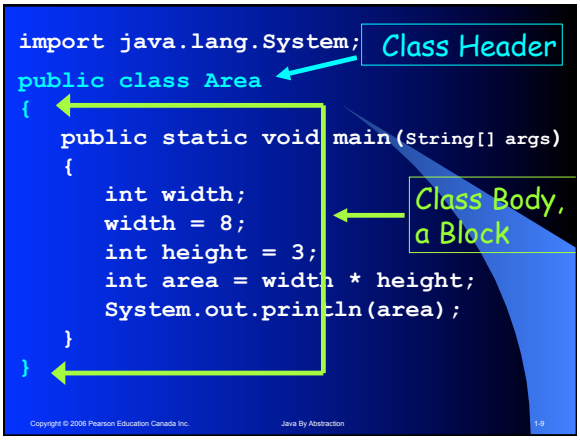


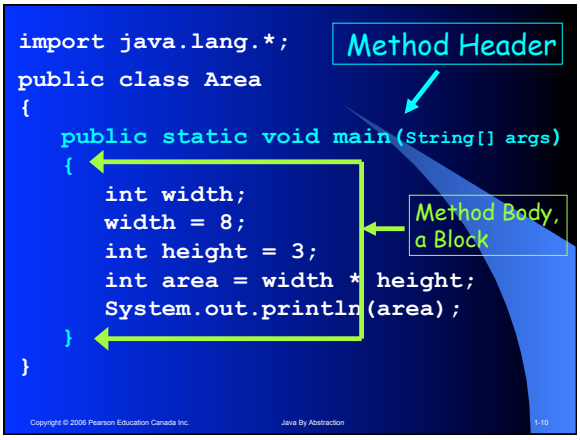


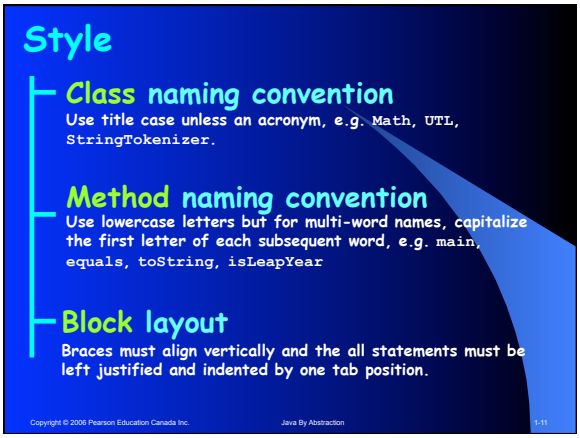


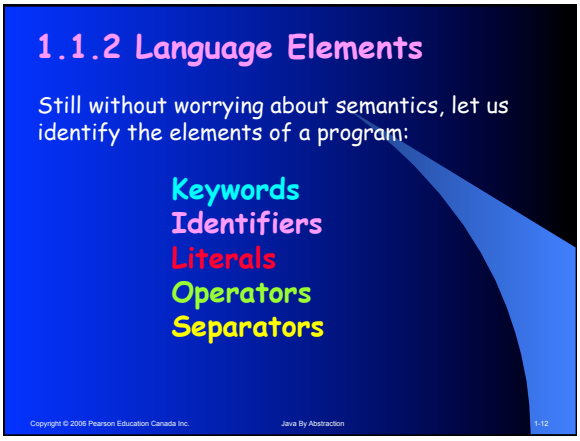


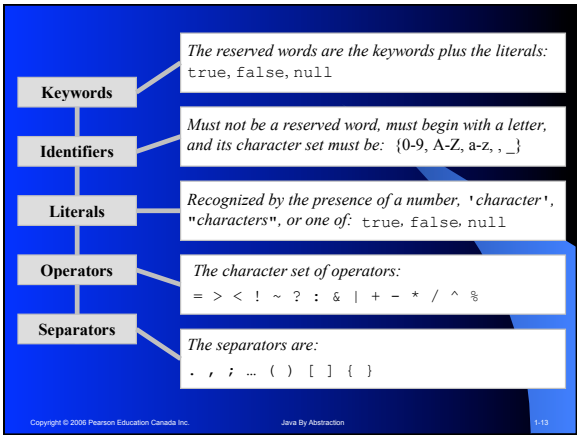


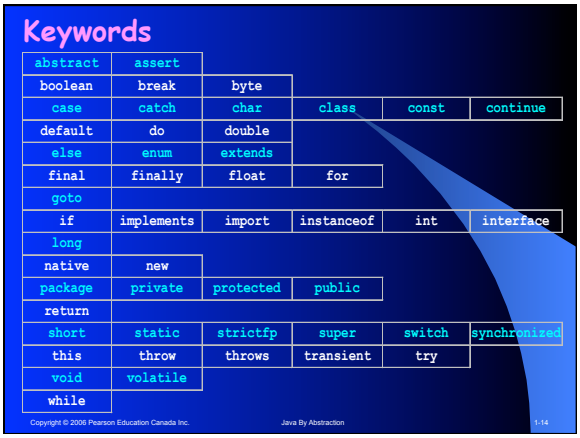


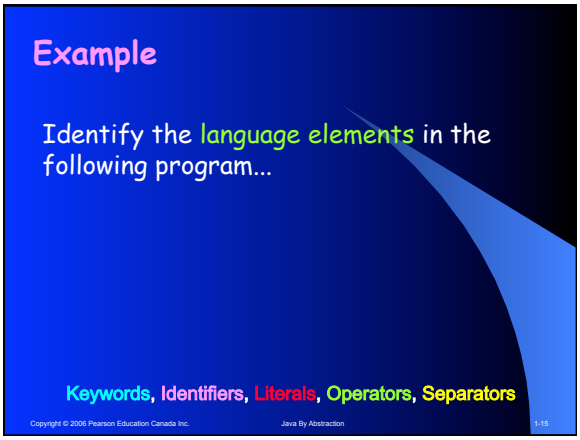

import java.lang.*;
Method Header
public class Area
{
public static void main(String[] args)
{
int width;
width = 8;
int height = 3;
int area = width * height;
System.out.println(area);
}
}
Method Body, a Block
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-10

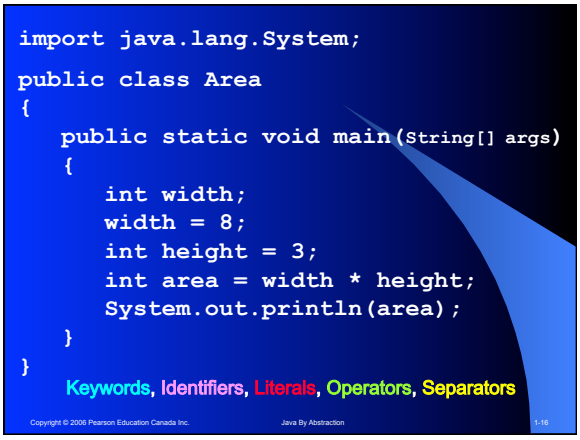

Style
Class naming convention
Use title case unless an acronym, e.g. Math, UTL, StringTokenizer.
Method naming convention
Use lowercase letters but for multi-word names, capitalize the first letter of each subsequent word, e.g. main, equals, toString, isLeapYear
Block layout
Braces must align vertically and the all statements must be left justified and indented by one tab position.
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-11

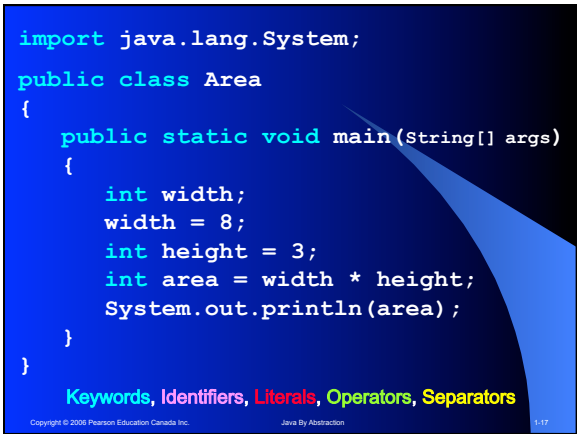

1.1.2 Language Elements
Still without worrying about semantics, let us identify the elements of a program:
Keywords
Identifiers
Literals
Operators
Separators
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-12

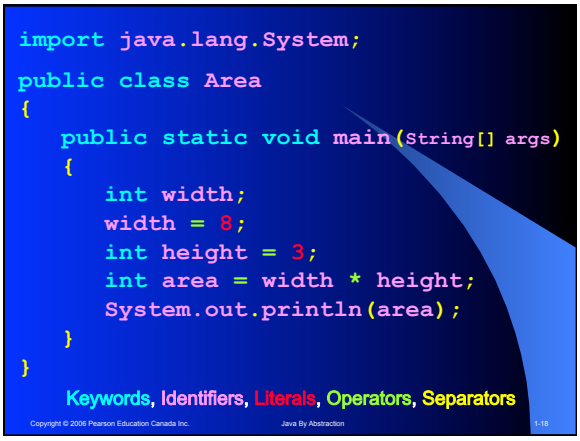

Keywords
Identifiers
Literals
Operators
Separators
The reserved words are the keywords plus the literals: true, false, null
Must not be a reserved word, must begin with a letter, and its character set must be: {0-9, A-Z, a-z, , _}
Recognized by the presence of a number, 'character', "characters", or one of: true, false, null
The character set of operators:
= > < ! ~ ? : & | + - * / ^ %
The separators are:
. , ; … ( ) [ ] { }
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-13


Keywords
abstract assert
boolean break byte
case catch char class const continue
default do double
else enum extends
final finally float for
goto
if implements import instanceof int interface
long
native new
package private protected public
return
short static strictfp super switch synchronized
this throw throws transient try
void volatile
while
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-14

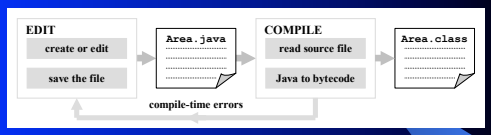

Example
Identify the language elements in the following program...
Keywords, Identifiers, Literals, Operators, Separators
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-15

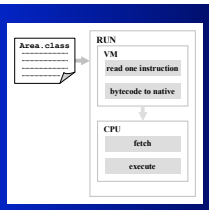






1.1.3 Program Execution





 Java By Abstraction 1-19

1.2 The Declaration Statement

```
type name;
```

The name of a primitive or non-primitive type, e.g. int, double…

A separator

An identifier to be associated with a memory block

The scope of the variable = the enclosing block of the declaration. The variable is not known outside its scope.

 Java By Abstraction 1-20

1.2.1 Variable Names

Rules and guidelines for the name:

- Must be an identifier
- Must not be in the scope of another variable with the same name
- A good name is indicative of the content that will be stored in the variable
- As a matter of style, use lowercase letters, but for multi-word names, capitalize the first letter of each subsequent word

 Java By Abstraction 1-21

1.2.2 The Integer Types

A type is a range of values and a set of operations on these values.

The range of the int type consists of all whole numbers between -2 and +2 billions (approx). int supports the four arithmetic operations plus the remainder.

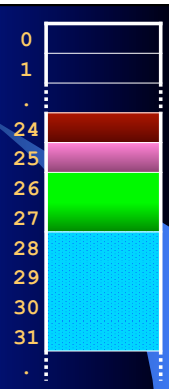
The long type is very similar to int except its range is much bigger, $+/-10^{19}$

An integer literal has an int type unless suffixed by L (l), in which case it is long.

 Java By Abstraction 1-22

1.2.3 Declaration and Memory

- Logical versus Physical
- Address versus Content
- Bytes, KB, MB, GB, TB, …
- The Memory Block
  - 1-byte block at address 24
  - 1-byte block at address 25
  - 2-byte block at address 26
  - 4-byte block at address 28



 Java By Abstraction 1-23

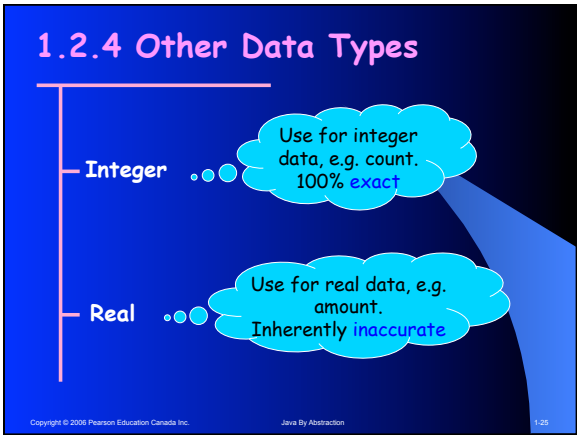
What happens when we write:

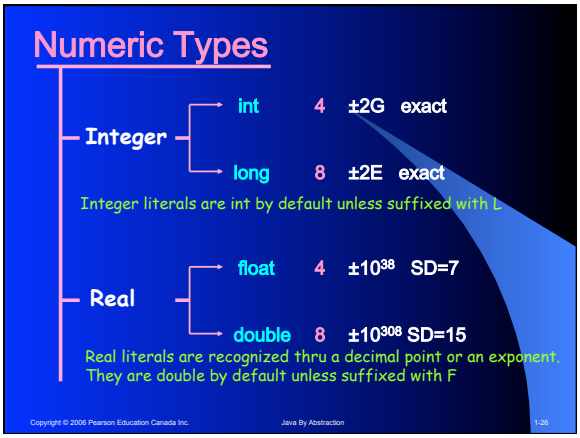
```
int width;
```

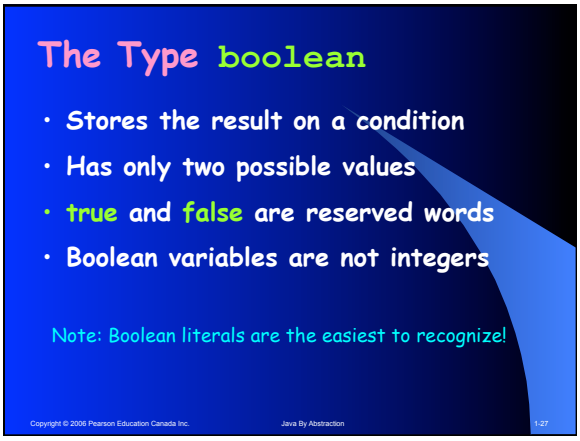
1. A block big enough to hold an int is allocated, e.g. a 4B block at 24
2. Its address is associated with the variable name, e.g. 24 with width

Note that no initialization is involved; only an association of a name with an address.

 Java By Abstraction 1-24


1.2.4 Other Data Types
Integer
Use for integer data, e.g. count. 100% exact
Real
Use for real data, e.g. amount. Inherently inaccurate
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-25


Numeric Types
Integer
int 4 ±2G exact
long 8 ±2E exact
Integer literals are int by default unless suffixed with L
Real
float 4 ±10^38 SD=7
double 8 ±10^308 SD=15
Real literals are recognized thru a decimal point or an exponent. They are double by default unless suffixed with F
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-26


The Type boolean
• Stores the result on a condition
• Has only two possible values
• true and false are reserved words
• Boolean variables are not integers
Note: Boolean literals are the easiest to recognize!
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-27

The Character Type `char`

- A letter, digit, or symbol
- Digits versus Numbers
- Store the code, not the typeface
- The case of English: ASCII
- `char` is thus an (unsigned) integer type
- Unicode has 64K codes

Character **literals** are recognized by single quotes surrounding one character, e.g. 'A'

 Java By Abstraction 1-28

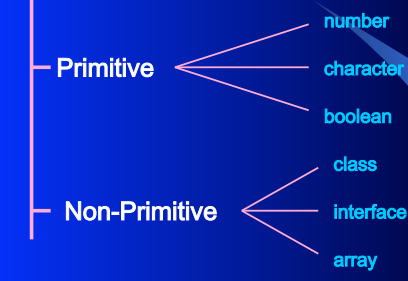
More on Characters

Code	Character
0	
⋮	
32	space
⋮	
48-57	'0'-'9'
⋮	
65-90	'A'-'Z'
⋮	
97-122	'a'-'z'
⋮	
65535	

Escape	Meaning
\uxxxx	The character whose code is (hex) xxxx
\'	Single quote
\"	Double quote
\\	Backslash
\n	New line
\r	Carriage return
\f	Form Feed
\t	Tab
\b	Backspace

 Java By Abstraction 1-29

1.2.5 Primitive & Non-Primitive



 Java By Abstraction 1-30

Java's Primitive Type

PRIMITIVE TYPES			Type	Size (bytes)	Approximate Range min	Approximate Range max	S.D.
NUMBER	INTEGER	SIGNED	byte	1	-128	+127	N/A
			short	2	-32,768	+32,767	N/A
			int	4	-2×10^{9}	$+2\times10^{9}$	N/A
			long	8	-9×10^{18}	$+9\times10^{18}$	N/A
		UNSIGNED	char	2	0	65,535	N/A
	REAL	SINGLE	float	4	$+3.4\times10^{38}$	$+3.4\times10^{38}$	7
		DOUBLE	double	8	-1.7×10^{308}	$+1.7\times10^{308}$	15
BOOLEAN			boolean	1	true/false		N/A

 Java By Abstraction 1-31

1.3 The Assignment Statement

An operator

```
name = value;
```

- Pre-declared and in-scope
- Its type can hold RHS
- Its content will be overwritten

A separator

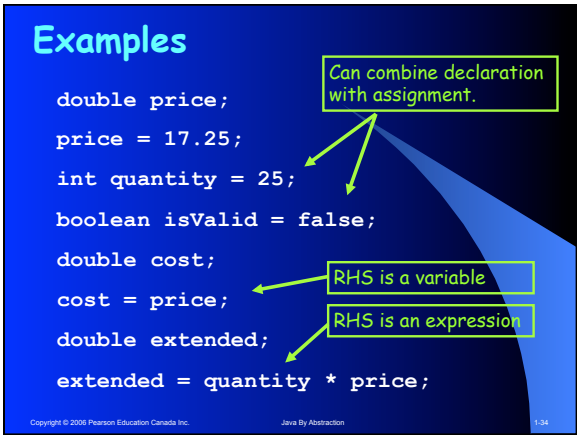
- a Literal
- a Name, or
- an Expression

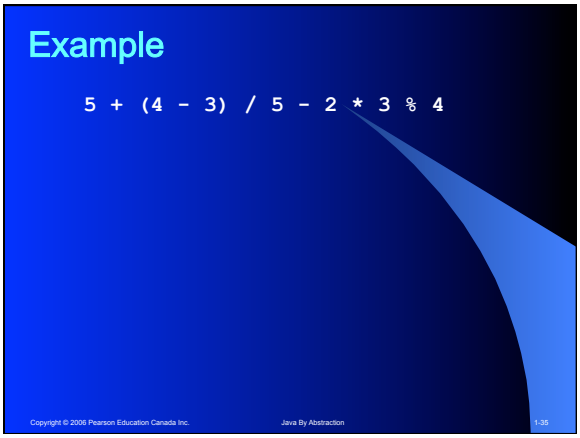
 Java By Abstraction 1-32

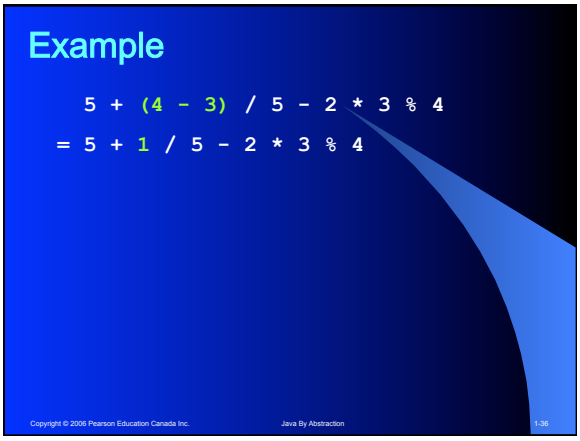
1.3.1 The `int` Arithmetic Operators

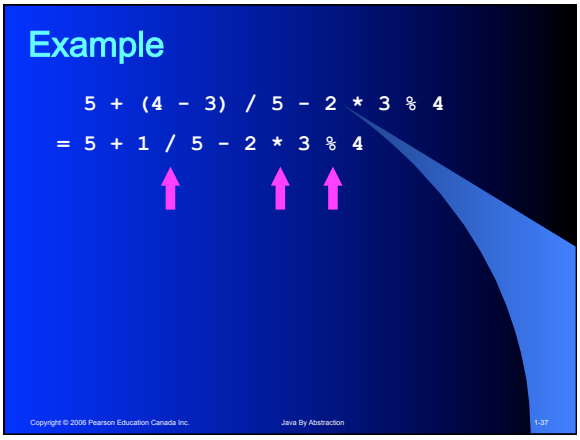
Precedence	Operator	Kind	Syntax	Operation
-5 →	+	infix	x + y	add y to x
	-	infix	x - y	subtract y from x
-4 →	*	infix	x * y	multiply x by y
	/	infix	x / y	divide x by y
	%	infix	x % y	remainder of x / y
-2 ←	+	prefix	+x	identity
	-	prefix	-x	negate x
	++	prefix	++x	x = x + 1; result = x
	--	prefix	--x	x = x - 1; result = x
-1 →	++	postfix	x++	result = x; x = x + 1
	--	postfix	x--	result = x; x = x - 1

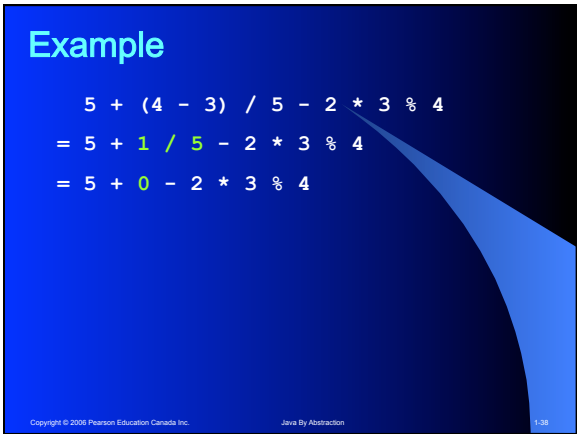
 Java By Abstraction 1-33

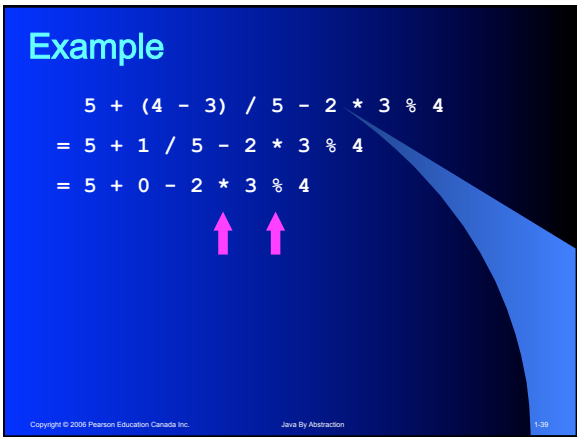

Examples
Can combine declaration with assignment.
double price;
price = 17.25;
int quantity = 25;
boolean isValid = false;
double cost;
cost = price;
RHS is a variable
double extended;
extended = quantity * price;
RHS is an expression
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-34

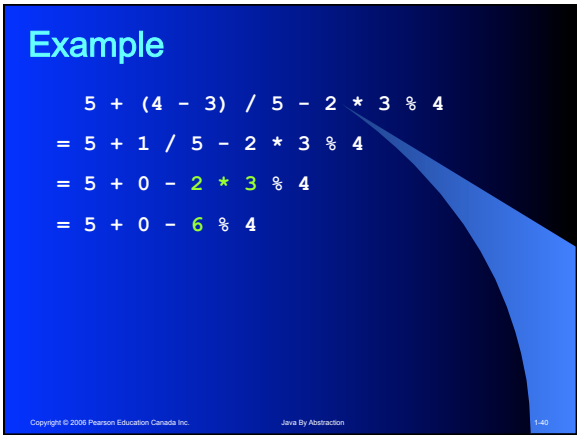

Example
5 + (4 - 3) / 5 - 2 * 3 % 4
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-35

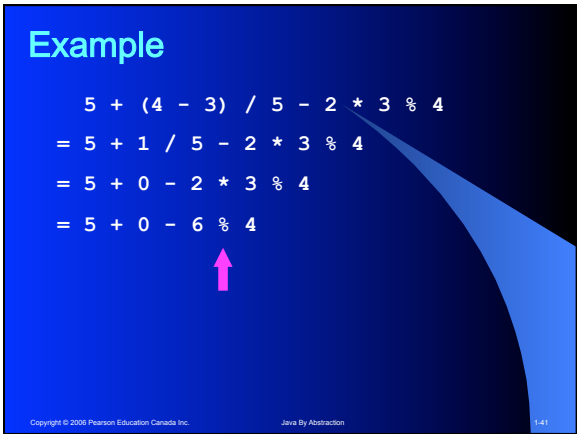

Example
5 + (4 - 3) / 5 - 2 * 3 % 4
= 5 + 1 / 5 - 2 * 3 % 4
Copyright © 2006 Pearson Education Canada Inc.
Java By Abstraction
1-36

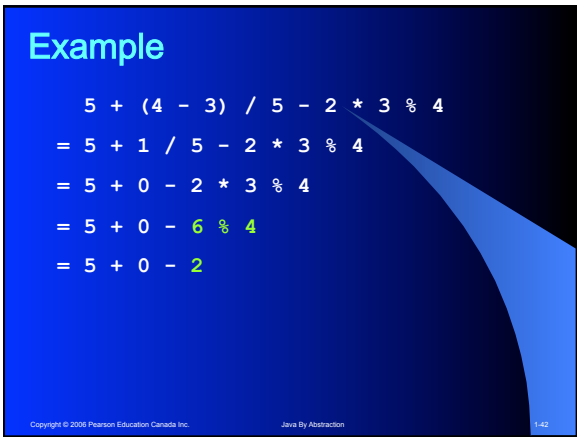


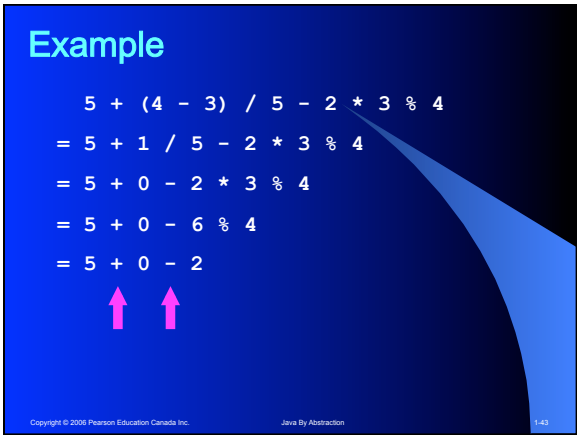


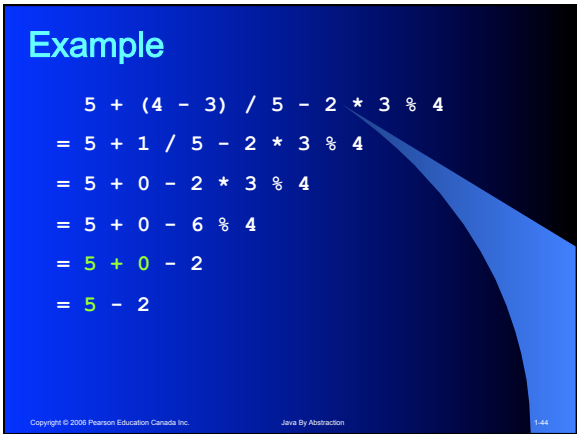


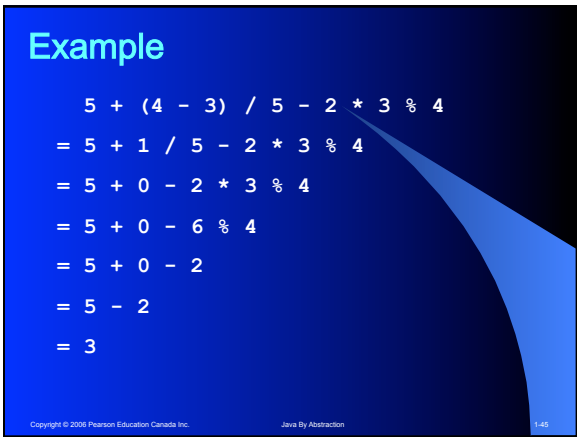












1.3.2 Other Arithmetic Operators

Each of long, float, and double come with 11 operators with the same symbols as int; i.e. the symbols are overloaded. Note:

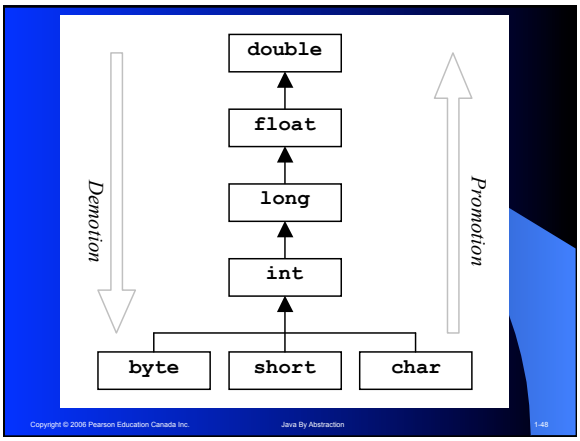
- The `int` operators satisfy closure thru circular wrapping
- The / int operator always rounds toward 0 and leads to an exception if the divisor is zero
- The sign of % is the same as that of the dividend
- The real operators satisfy closure by adding `Infinity` and `NaN`. Hence, dividing by zero does not lead to exceptions
- `(a * b) / c` is not the same as `a * (b / c)` for any type
- `(a + b) - c` is not the same as `a + (b - c)` for real types

 Java By Abstraction 1-46

1.3.3 Mixed Types and Casting

- Promotion (aka widening conversion) is done automatically when needed
- May lead to loss of precision but the order of magnitude is preserved
- Demotion is not done automatically. Can be done manually thru a cast
- Casting is risky…avoid it.

 Java By Abstraction 1-47



 Java By Abstraction 1-48

Note:

- The cast operator has a precedence that is higher than * but less than ++
- The = operator has the lowest precedence of all operators
- There are shorthand operators to combine assignment with an operator:

`x op= y` is shorthand for `x = x op y`

Ex: `x +=1` is like `x = x + 1` or `x++`

 Java By Abstraction 1-49

Example

```
int iVar = 15;
long lVar = 2;
float fVar = 7.6f - iVar / lVar;
double dVar = 1L / lVar + fVar / lVar;
int result = 100 * dVar;
```

Fix, if need be, and output `result`
The answer may surprise you!

 Java By Abstraction 1-50