Assignment Statements and Arithmetic

Objectives
Topics

- Type compatibility.
- Assignment operators.
- Arithmetic.

Type Compatibility

- All of the primitive types except boolean and char are compatible with each other.
  - char type is compatible with everything except boolean, byte, and short
  - boolean type is not compatible with anything else
- Class types are compatible if one is a subclass of the other.
- The value assigned to a variable must have the same type as the variable, or be a narrower, compatible type.
Compatible Primitive Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Bits</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>byte</td>
<td>8</td>
<td>-128 to +127</td>
</tr>
<tr>
<td>short</td>
<td>16</td>
<td>-32,768 to +32,767</td>
</tr>
<tr>
<td>char†</td>
<td>16</td>
<td>0 to 65,535</td>
</tr>
<tr>
<td>int</td>
<td>32</td>
<td>± 2.1 billion (approximate)</td>
</tr>
<tr>
<td>long</td>
<td>64</td>
<td>± 9.2e18 (approx.)</td>
</tr>
<tr>
<td>float</td>
<td>32</td>
<td>± 3.4e38 (~7 digit precision)</td>
</tr>
<tr>
<td>double</td>
<td>64</td>
<td>± 1.8e308 (~10 digit precision)</td>
</tr>
</tbody>
</table>

†Not compatible with byte or short

Compatible Classes

```
Animal
  | Mammal
  |  | Carnivore
  |  |  | Canine
  |  |  |  | Wolf
```

```
Object
  | Component
  |  | Container
  |  |  | JComponent
  |  |  |  | AbstractButton
  |  |  |  |  | JMenuItem
  |  |  |  |  |  | JMenu
```
Assignment Examples

```java
type n;
char c;
double d;
;
int n; // OK
c = n; // error
d = n; // OK
d = c; // OK
c = d; // error

JMenuItem item = new JMenuItem();
JMenu menu = new JMenu();

item = menu; // OK
menu = item; // error

Object obj = menu; // OK
item = obj; // error: even though
    // obj holds a menu
```

Type Conversions

- When a value of a primitive type is assigned to a variable of a wider type, the value is converted.
  - That is, its binary encoding is changed.
- When an object is assigned to a variable of a wider type it is not converted.
  - Since object variables hold references.
Literal and Expression Types

- Variables, literals, and expressions all have types.
- A character in single quotes has type `char`:
  - ‘A’, ‘3’, ‘g’, ‘$’, ‘+’, ...
- A number without a decimal point or exponent has type `int`:
  - 5, 3, 23849, ...
- A number with a decimal point or exponent has type `double`:
  - 5.0, 3.14159, 450.2172, 4e2, 12.467e18
- To write a literal of type long, or float add a L, or F (or l or f):
  - 94L, 12.4f
- There is no way to write a literal of type byte or short.
  - But you can assign small literals to byte and short variables.
- The type of an expression is the type of value it produces when it’s evaluated.

Arithmetic Operators

- Operators
  - addition: +
  - subtraction: -
  - multiplication: *
  - division: /
  - modulus: %
Arithmetic Operations and Types

- Arithmetic is never performed on types narrower than int.
  - If an operand is narrower than int, it is widened for use in the operation.
- Arithmetic is never performed on mixed types.
  - If the types of the operands don’t match the narrower operand is widened to match the wider operand.
- If the operands are integers (int or long) integer arithmetic is performed, otherwise (float or double) floating point arithmetic is performed.
  - The result has the same type as the operands (after any widening).
  - Integer division truncates (chops off) any remainder.

Precedence and Associativity in Arithmetic Expressions

- Operator Precedence (highest to lowest)
  1. parentheses
  2. multiplication, division, and modulus operators
  3. addition and subtraction operators
  4. assignment operator
- Associativity
  - All associate from left to right except assignment
  - Assignment operators associate right to left
- These are the same rules you learned in 7th grade mathematics (except for the assignment operator).
## Arithmetic Examples

```java
short s1 = 1, s2 = 3;
int i = 4, j = 5, k = 20;
float f = 2.0;
double d = 4.0;
```

<table>
<thead>
<tr>
<th>Expression</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>i + j</td>
<td>int</td>
<td>9</td>
</tr>
<tr>
<td>s1 + s2</td>
<td>int</td>
<td>4</td>
</tr>
<tr>
<td>k - f</td>
<td>float</td>
<td>18.0</td>
</tr>
<tr>
<td>i / j</td>
<td>int</td>
<td>0</td>
</tr>
<tr>
<td>d / j</td>
<td>double</td>
<td>0.8</td>
</tr>
<tr>
<td>i - j + 3</td>
<td>int</td>
<td>2</td>
</tr>
<tr>
<td>i - (j + 3)</td>
<td>int</td>
<td>-4</td>
</tr>
<tr>
<td>i + j * 3</td>
<td>int</td>
<td>19</td>
</tr>
<tr>
<td>(i + j) * 3</td>
<td>int</td>
<td>27</td>
</tr>
<tr>
<td>d + i / j</td>
<td>double</td>
<td>4.0</td>
</tr>
<tr>
<td>d / j + i</td>
<td>double</td>
<td>4.8</td>
</tr>
</tbody>
</table>

## Beware of Error Propagation

- Mathematically, the following are equivalent:
  \[
  \left(\frac{a \cdot b}{c}\right) = \left(\frac{a \cdot b}{c}\right) = \left(\frac{a \cdot b}{c}\right) 
  \]
  if \( a = 5, b = 25, c = 10 \)
  then the value is 12.5

- In java, the results are different:
  
  \((5*25)/10 \Rightarrow 12\)
  
  \(5 \times (25/10) \Rightarrow 10\)
  
  \((5/10)*25 \Rightarrow 0\)
Modulus Operator

- The modulus operator (%) gives the remainder after integer division.
  
  \[ 12 \mod 3 \Rightarrow 0 \]
  
  \[ 12 \mod 5 \Rightarrow 2 \]

Assignment Expressions

- Assignments are expressions whose value is the value that is assigned.
- Assignment operator associates from right to left.

```c
int i = 0, j = 0, k = 0;

i = j = k = 10; // assign all variables the value 10
```
More Assignment Operators

- +=, -=, *=, /=, %=
- An expression:
  var ?= expression
  where ? is +, -, *, /, or %
- Is equivalent to:
  var = var ? (expression)
- Examples:
  x += 5  means same as x = x + 5
  y *= x + 2 means same as y = y * (x + 2)

Type Casts

- Type casts can used to force a value into a different type. This is dangerous!!

```java
byte b = 0;
int i = 1000;
b = i; // error: can’t assign an int to a byte
b = (byte)i; // OK, but what is the result?
```

A type cast