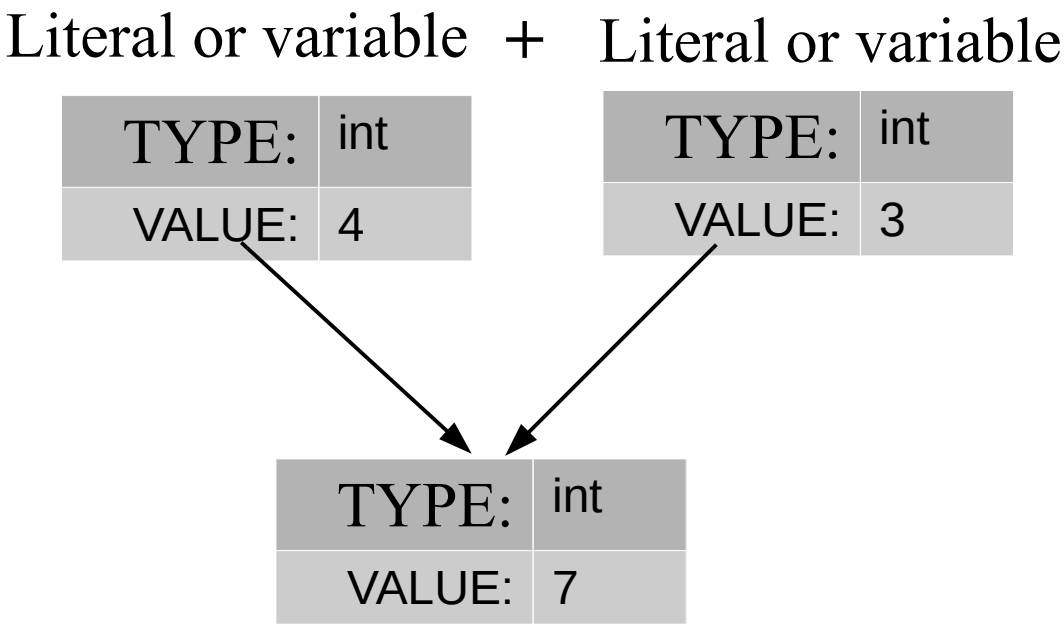
BINARY ARITHMETIC OPERATORS



Same idea with the other basic operators: -, *, /, %

How is the type computed?

ARITHMETIC OPERATOR RESULT TYPE

- a **double** if at least one of the operands is a double
- otherwise, a **float** if at least one of the operands is a float
- otherwise, a long if at least one of the operands is a long
- otherwise, an **int** even in both operands are byte, short, or char

EXAMPLE:

```
int x=3;
int y=4;

System.out.println(x+y);
System.out.println(y-x);
System.out.println(x*y);
System.out.println(y/x);
System.out.println(y/x);
System.out.println(y%x);
```

Printout:

```
7
1
12
1
0
1
```

In this example ALL RESULTS ARE **int** (see the rules above)

ARITHMETIC EXPRESSIONS

- Results in a numeric value and type
- Uses arithmetic operators (could be non-binary… more about those later)
- Use variables and/or literals
- Follow operator precedence
- Can use parenthesis to modify operator precedence

ARITHMETIC OPERATOR PRECEDENCE:

1) Parentheses (inner first)
2) Multiplication, division, modulus (from left to right)
3) Addition and subtraction (from left to right)

EXAMPLES OF ARITHMETIC EXPRESSIONS

```
int a=3;          int b=5;
int c=7;          int d=2;
double x=3.8;     double y=2;
double z=5.4;
```

Printout:

```
System.out.println(a+b*c);
System.out.println(a+b*c+a*c);
System.out.println((a+b)*c);
System.out.println(b/d);
System.out.println(b/y);
System.out.println(c/3);
System.out.println(z/3);
System.out.println(x*x);
```

```
38
59
56
2
2.5
2
1.8
14.44
```

In class do all the diagrams of type-value computation

THE ASSIGNMENT OPERATOR

<VARIABLE> = <EXPRESSION> ;

The = (equal) symbol is used to

ASSIGN THE VALUE OBTAINED AFTER COMPUTING THE EXPRESSION TO THE VARIABLE ON THE LEFT HAND SIDE

RULES:

- Left hand side MUST be a variable
- Right hand side MUST be an expression

The semantics (meaning) of the = in Java is not the same as in mathematics (equations).

For example,

i=i+1

Increments the value of i by 1 and assigns it to i

EXAMPLE

```
int x=3;
int y=4;
int z;
double w;

z=x*x+y*y;
System.out.println(z);

w=(z-x)*y
System.out.println(w);

x=x+1;
System.out.println(x);
```

In pseudocode the assignment operator is many times represented with an arrow:

$x \leftarrow 3.5+y$

Printout:

```
25
88.0
4
```

Booleans

- The result of a comparison
- Can be used for combining tests
  - Hour value is invalid if hour < 0 or hour > 12
- A boolean value represents the result of a testing condition
- There are only two values
  - true or false
- Like other data booleans are stored as variables

Boolean values as data

- Since true and false are data value, we can:
  - Store them in variables
  - Read them from input and print as output
  - Create them with operations

    x<y x>y x<=y x>=y x==y x!=y
  - Use them as operand for boolean operators

    xBig || yBig xBig && yBig !xBig

Boolean Operators: or, and, not

A	B	A || B
false	false	**false**
false	true	**true**
true	false	**true**
true	true	**true**

A	B	A && B
false	false	**false**
false	true	**false**
true	false	**false**
true	true	**true**

A	!A
false	**true**
true	**false**

Boolean examples

x ← 10, y ← 20

- x < y
- x >= y
- x >= 10
- x == 9 || x == 10
- x == 9 && x == 10
- x > y || (x == 10 && y < 7)