Lecture 2: Operations and Data Types

Building Java Programs: A Back to Basics Approach
by Stuart Reges and Marty Stepp

Data types

- **type**: A category or set of data values.
  - Constrains the operations that can be performed on data
  - Many languages ask the programmer to specify types
  - Examples: integer, real number, string

- Internally, computers store everything as 1s and 0s

```
104    → 01101000
"hi"   → 0110100 0110101
```

Java's primitive types

- **primitive types**: 8 simple types for numbers, text, etc.
  - Java also has **object types**, which we'll talk about later

Name	Description		Examples
`int`	**integers**	(up to $2^{31} - 1$)	`42, -3, 0, 926394`
`double`	**real numbers**	(up to 10^{38})	`3.1, -0.25, 9.4e3`
`boolean`	`logical values`		`true, false`

Expressions

- **expression**: A value or operation that computes a value.
  - Examples:

```
1 + 4 * 5
(7 + 2) * 6 / 3
42
```

  - The simplest expression is a *literal value*.
  - A complex expression can use operators and parentheses.

Arithmetic operators

- **operator**: Combines multiple values or expressions.

`+`	addition
`-`	subtraction (or negation)
`*`	multiplication
`/`	division
`%`	modulus (a.k.a. remainder)

- As a program runs, its expressions are *evaluated*.
  - `1 + 1` evaluates to `2`
  - `System.out.println(3 * 4);` prints `12`
    - How would we print the text `3 * 4` ?

Integer division with /

- When we divide integers, the quotient is also an integer.
  - `14 / 4` is `3`, not `3.5`

```
    3
4 ) 14
    12
     2
```

```
    4
10 ) 45
    40
     5
```

```
     52
27 ) 1425
    135
     75
     54
     21
```

- More examples:
  - `32 / 5` is `6`
  - `84 / 10` is `8`
  - `156 / 100` is `1`

  - Dividing by 0 causes an error when your program runs.

Integer remainder with %

- The % operator computes the remainder from integer division.
  - `14 % 4` is `2`
  - `218 % 5` is `3`

```
      3
  4 ) 14
      12
       2
```

```
     43
  5 ) 218
      20
       18
       15
        3
```

What is the result?

```
45 % 6
2 % 2
8 % 20
11 % 0
```

- Applications of % operator:
  - Obtain last digit of a number: `230857 % 10` is `7`
  - Obtain last 4 digits: `658236489 % 10000` is `6489`
  - See whether a number is odd: `7 % 2` is `1`, `42 % 2` is `0`

Expressions

Find the exact change for 137 cents using quarters, dimes, nickels and cents. Use the least number of coins.

How many quarters? 137 / 25 = 5 quarters (Integer Division!)

What's leftover? 137 % 25 = 12 cents

How many dimes? 12 / 10 = 1 dime

What's leftover? 12 % 10 = 2 cents

How many nickels? 2 / 5 = 0 nickels.

What's leftover? 2 % 5 = 2 cents.

Precedence

- **precedence**: Order in which operators are evaluated.
  - Generally operators evaluate left-to-right.
    `1 - 2 - 3` is `(1 - 2) - 3` which is `-4`
  - But `* / %` have a higher level of precedence than `+ -`

```
1 + 3 * 4              is 13

6 + 8 / 2 * 3
6 +   4   * 3
6 +     12             is 18
```

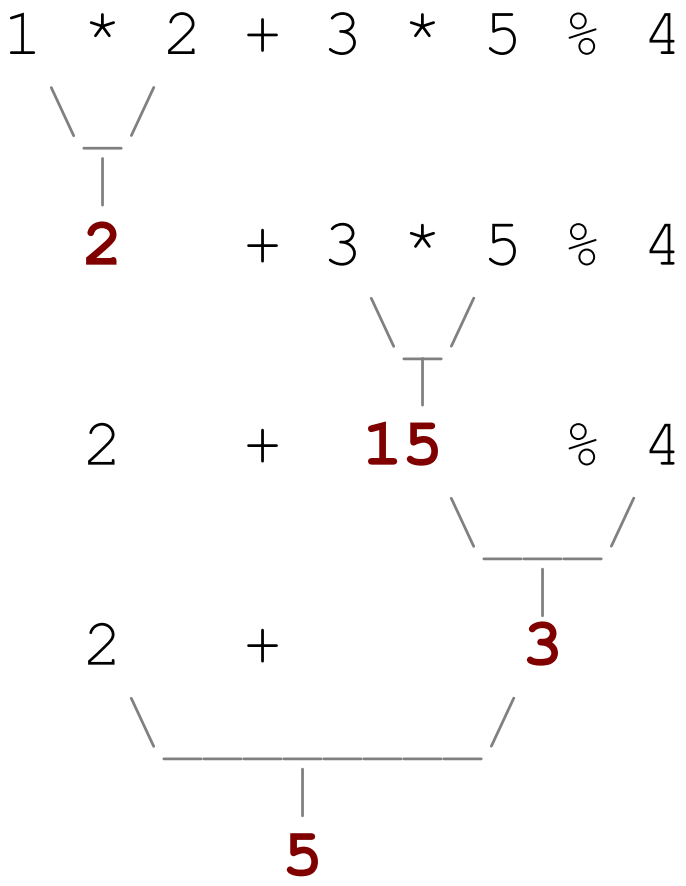
  - Parentheses can force a certain order of evaluation:

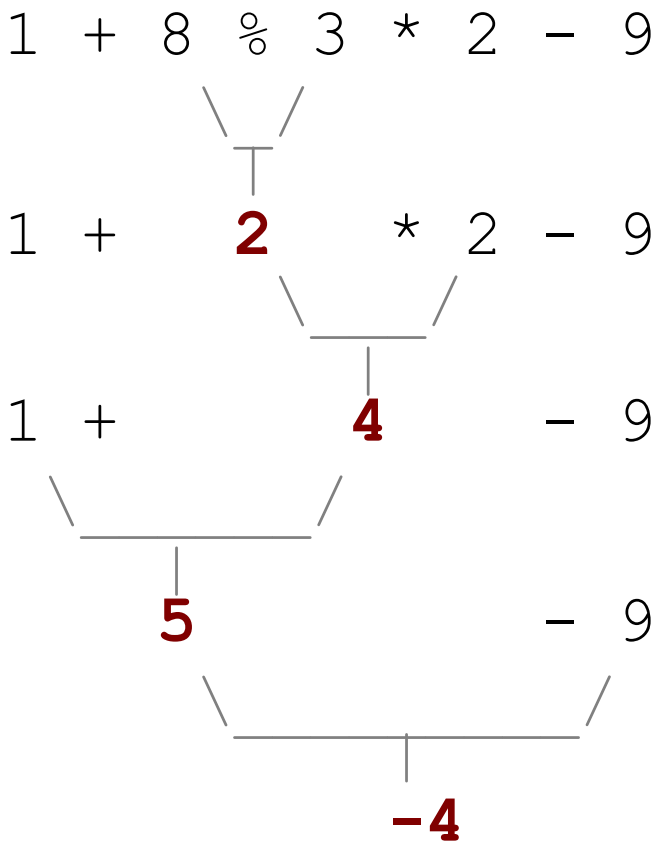
```
(1 + 3) * 4            is 16
```

  - Spacing does not affect order of evaluation

```
1+3 * 4-2              is 11
```

Precedence examples

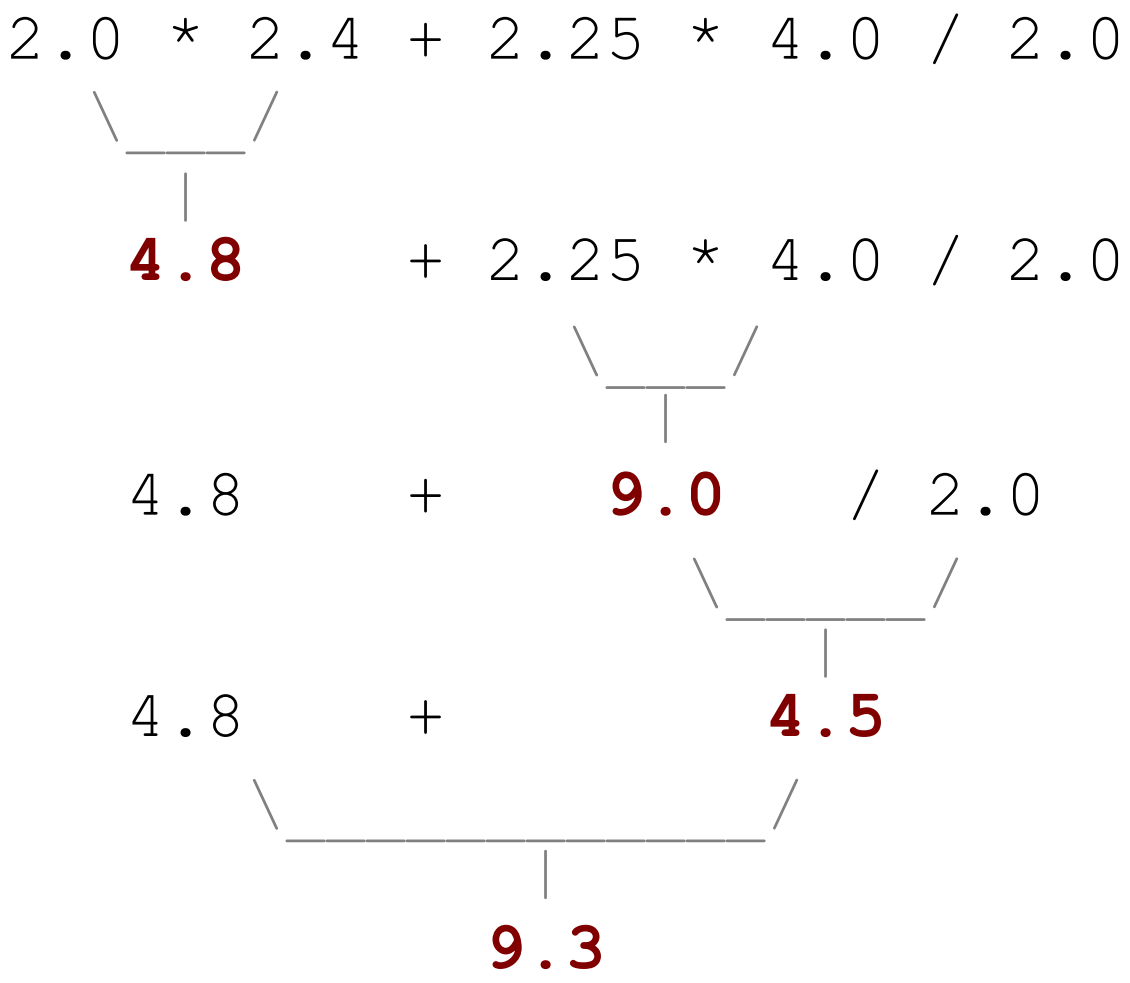




Real numbers (type `double`)

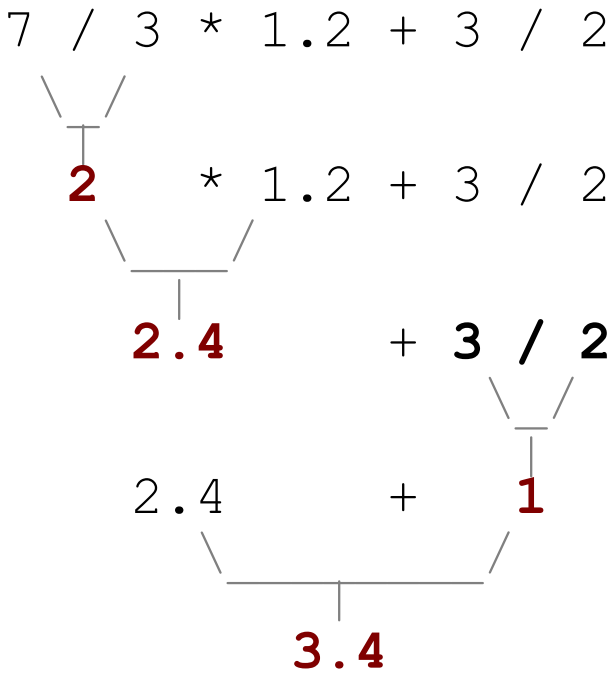
- Examples: `6.022, -42.0, 2.143`
  - Placing `.0` or `.` after an integer makes it a `double`.
- The operators `+ - * / % ()` all still work with `double`.
  - `/` produces an exact answer: `15.0 / 2.0` is `7.5`
  - Precedence is the same: `()` before `* / %` before `+ -`

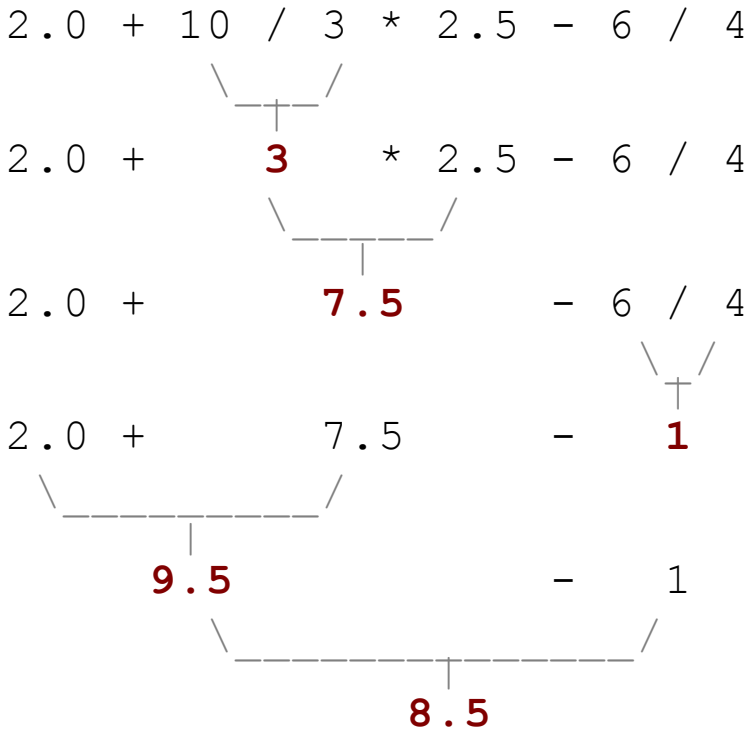
Real number example



Mixing types

- When `int` and `double` are mixed, the result is a `double`.
  - `4.2 * 3` is `12.6`
- The conversion is per-operator, affecting only its operands.





  - `3 / 2` is `1` above, not `1.5`.

String concatenation

- **string concatenation**: Using + between a string and another value to make a longer string.

```
"hello" + 42    is   "hello42"
1 + "abc" + 2   is   "1abc2"
"abc" + 1 + 2   is   "abc12"
1 + 2 + "abc"   is   "3abc"
"abc" + 9 * 3   is   "abc27"
"1" + 1         is   "11"
4 - 1 + "abc"   is   "3abc"
```

- Use + to print a string and an expression's value together.
  - `System.out.println("Grade: " + (95.1 + 71.9) / 2);`
  - Output: `Grade: 83.5`

Variables

Receipt example

What's bad about the following code?

```
public class Receipt {
    public static void main(String[] args) {
        // Calculate total owed, assuming 8% tax / 15% tip
        System.out.println("Subtotal:");
        System.out.println(38 + 40 + 30);
        System.out.println("Tax:");
        System.out.println((38 + 40 + 30) * .08);
        System.out.println("Tip:");
        System.out.println((38 + 40 + 30) * .15);
        System.out.println("Total:");
        System.out.println(38 + 40 + 30 +
                           (38 + 40 + 30) * .08 +
                           (38 + 40 + 30) * .15);
    }
}
```

- The subtotal expression `(38 + 40 + 30)` is repeated
- So many `println` statements

Variables

- **variable**: A piece of the computer's memory that is given a name and type, and can store a value.
  - Like preset stations on a car stereo, or cell phone speed dial:

  - Steps for using a variable:
    - *Declare* it - state its name and type
    - *Initialize* it - store a value into it
    - *Use* it - print it or use it as part of an expression

Declaration

- **variable declaration**: Sets aside memory for storing a value.
  - Variables must be declared before they can be used.

- Syntax:

  type name;

  - The name is an *identifier*.

  - `int x;`

  - `double myGPA;`

Assignment

- **assignment**: Stores a value into a variable.
  - The value can be an expression; the variable stores its result.

- Syntax:

```
name = expression;
```

  - ```
    int x;
    x = 3;
    ```

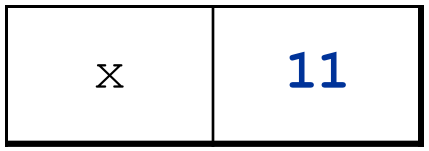

  - ```
    double myGPA;
    myGPA = 1.0 + 2.25;
    ```

Using variables

- Once given a value, a variable can be used in expressions:

```
int x;
x = 3;
System.out.println("x is " + x);      // x is 3
System.out.println(5 * x - 1);        // 14
```

- You can assign a value more than once:




```
int x;
x = 3;
System.out.println(x + " here");      // 3 here

x = 4 + 7;
System.out.println("now x is " + x); // now x is 11
```

Declaration/initialization

- A variable can be declared/initialized in one statement.

- Syntax:

```
type name = value;
```

  - `double myGPA = 3.95;`

myGPA	3.95

  - `int x = (11 % 3) + 12;`

x	14

Assignment and algebra

- Assignment uses = , but it is not an algebraic equation.

  = means, *"store the value at right in variable at left"*

  - The right side expression is evaluated first, and then its result is stored in the variable at left.

- What happens here?

```
int x = 3;
x = x + 2;    // ???
```

Assignment and types

- A variable can only store a value of its own type.
  - `int x = 2.5;` `// ERROR: incompatible types`

- An `int` value can be stored in a `double` variable.
  - The value is converted into the equivalent real number.
  - `double myGPA = 4;`

myGPA	4.0

  - `double avg = 11 / 2;`

avg	**5.0**

    - Why does `avg` store `5.0` and not `5.5` ?

Compiler errors

- Order matters.
  - `int x;`

    `7=x; // ERROR: should be x=7;`
- A variable can't be used until it is assigned a value.
  - `int x;`

    `System.out.println(x);  // ERROR: x has no value`
- You may not declare the same variable twice.
  - `int x;`
    `int x;  // ERROR: x already exists`
  - `int x = 3;`
    `int x = 5;  // ERROR: x already exists`
    - How can this code be fixed?

Printing a variable's value

- Use + to print a string and a variable's value on one line.

```
double grade = (95.1 + 71.9 + 82.6) / 3.0;
System.out.println("Your grade was " + grade);

int students = 11 + 17 + 4 + 19 + 14;
System.out.println("There are " + students +
                   " students in the course.");
```

- Output:

```
Your grade was 83.2
There are 65 students in the course.
```

Receipt question

Improve the receipt program using variables.

```
public class Receipt {
    public static void main(String[] args) {
        // Calculate total owed, assuming 8% tax / 15% tip
        System.out.println("Subtotal:");
        System.out.println(38 + 40 + 30);

        System.out.println("Tax:");
        System.out.println((38 + 40 + 30) * .08);

        System.out.println("Tip:");
        System.out.println((38 + 40 + 30) * .15);

        System.out.println("Total:");
        System.out.println(38 + 40 + 30 +
                           (38 + 40 + 30) * .15 +
                           (38 + 40 + 30) * .08);
    }
}
```

Receipt answer

```
public class Receipt {
    public static void main(String[] args) {
        // Calculate total owed, assuming 8% tax / 15% tip
        int subtotal = 38 + 40 + 30;
        double tax = subtotal * .08;
        double tip = subtotal * .15;
        double total = subtotal + tax + tip;

        System.out.println("Subtotal: " + subtotal);
        System.out.println("Tax: " + tax);
        System.out.println("Tip: " + tip);
        System.out.println("Total: " + total);
    }
}
```

Flow of a Program

```
public class DrawBoxes2 {
    public static void main(String[] args) {
        drawBox();
        System.out.println();
        drawBox();
    }

    public static void drawBox() {
        System.out.println("+------+");
        System.out.println("|      |");
        System.out.println("|      |");
        System.out.println("+------+");
    }
}
```

Try

```
1  public class FooBarBazMumble {
2      public static void main(String[] args) {
3          foo();
4          bar();
5      }
6
7      public static void foo() {
8          System.out.println("foo");
9          mumble();
10         System.out.println();
11     }
12
13     public static void bar() {
14         System.out.println("bar");
15         baz();
16     }
17
18     public static void baz() {
19         System.out.println("baz");
20         mumble();
21     }
22
23     public static void mumble() {
24         System.out.println("mumble");
25     }
26 }
```

Output

- Output of FooBarBazMumble:

foo
mumble

bar
baz
mumble

Lab 1

Write a program from scratch. (NO COPY AND PASTE ALLOWED)

SAVE YOUR PROGRAM IN A FOLDER CALLED MATHEX ON CS50IDE.

You will write two static methods in addition to the main() method. Your methods should be called printMath() and printCat().

In the printMath() method, use both println and print commands to print some math expressions and check the evaluated value at runtime. You must use each of the operators `+ - * / % ()` and have at least five expressions.

In the printCat() method, use both println and print commands to print some examples of String concatenation. You must use both numbers and letters in your concatenation. Print at least five concatenations.

Lab 2

Write a program from scratch. (NO COPY AND PASTE ALLOWED)
SAVE YOUR PROGRAM IN A FOLDER CALLED ExactChange ON CS50IDE.

Use the following template(or something similar) to write a program that gives exact change with the least number of coins for a given number of cents. **Use intermediate variables to help your calculation.**

```
public static void main(String[] args){
   int totalCents = 137;  //137 can be any number
   ….. // your code here.
}
```

Output: 5 quarters, 1 dimes, 0 nickels, 2 pennies.

Lab 2

- Let {a1,a2,a3,…,an} be a list of n real numbers.
- The average of the list denoted by **ave** = (a1+a2+…+an)/n.
- The deviation of the entry ai from the average denoted by **di** = ai-ave.
- The variance of the list =
  [(d1)^2 + (d2)^2 + … + (dn)^2] / n.
- The standard deviation of the list = the square root of the variance of the list.

Lab 3

For example, if the list is {2,4,5,8,16}.

Average=7

d1=2-7=-5

d2=4-7=-3

d3=5-7=-2

d4=8-7=1

d5=16-7=9

Variance=[(-5)^2+(-3)^2+(-2)^2+1^2+9^2]/5=24

Standard deviation=square root of 24=4.898979486

Lab 3

Create a folder called Statistics to store your code.

- Create three integer variables to store three test scores: test1, test2, test3.
- Create an double variable to compute the average of the scores.
- Create three double variables to compute the deviations.
- Create two double variables to compute the variance and standard deviation.

- Hint: To compute the square root of 9 for example, use Math.sqrt(9).

Lab 3

- Your output should look something like:

The test scores are: 78, 82, 89.

The average of the scores is: 83.0

The variance is: 20.666666666666668

The standard deviation is: 4.546060565661952