



Math Return values & Scanner

Subset of the Supplement Lesson slides from: [Building Java Programs](http://www.buildingjavaprograms.com/), Chapter 3 Graphics
by Stuart Reges and Marty Stepp (<http://www.buildingjavaprograms.com/>)
& Thanks to Ms Martin for her suggestions too.

Distance between points

- Write a method that given x and y coordinates for two points prints the distance between them

If you can't do all of it, pseudocode?

What?! You don't remember the distance formula?!

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Java Libraries

- Java comes with lots of goodies
 - Graphics
 - Color
 - And Math, too! -> **java.lang.Math**
- Learn more from the Application Programming Interface linked from class page as “Java 6 API”:
<http://download.oracle.com/javase/6/docs/api/index.html>

Java's Math class

Method name	Description
<code>Math.abs(<i>value</i>)</code>	absolute value
<code>Math.ceil(<i>value</i>)</code>	rounds up
<code>Math.floor(<i>value</i>)</code>	rounds down
<code>Math.log10(<i>value</i>)</code>	logarithm, base 10
<code>Math.max(<i>value1</i>, <i>value2</i>)</code>	larger of two values
<code>Math.min(<i>value1</i>, <i>value2</i>)</code>	smaller of two values
<code>Math.pow(<i>base</i>, <i>exp</i>)</code>	<i>base</i> to the <i>exp</i> power
<code>Math.random()</code>	random double between 0 and 1
<code>Math.round(<i>value</i>)</code>	nearest whole number
<code>Math.sqrt(<i>value</i>)</code>	square root
<code>Math.sin(<i>value</i>)</code> <code>Math.cos(<i>value</i>)</code> <code>Math.tan(<i>value</i>)</code>	sine/cosine/tangent of an angle in radians
<code>Math.toDegrees(<i>value</i>)</code> <code>Math.toRadians(<i>value</i>)</code>	convert degrees to radians and back

Constant	Description
<code>Math.E</code>	2.7182818...
<code>Math.PI</code>	3.1415926...

No output?

- Simply calling these methods produces no visible result.
 - `Math.pow(3, 4); // no output`
- Math method calls use a Java feature called return values that cause them to be treated as expressions.
- The program runs the method, computes the answer, and then "replaces" the call with its computed result value.
 - ~~`Math.pow(3, 4);`~~ `// no output`
 - `81.0; // no output`
- To see the result, we must print it or store it in a variable.
 - `double result = Math.pow(3, 4);`
 - `System.out.println(result); // 81.0`

Calling Math methods

`Math.methodName (parameters)`

- Examples:

```
double squareRoot = Math.sqrt(121.0);  
System.out.println(squareRoot);           // 11.0
```

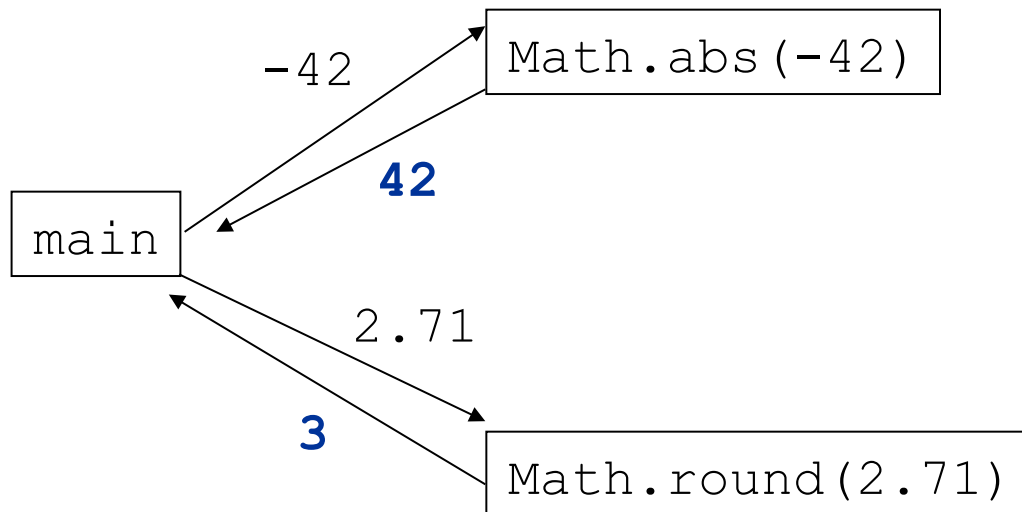
```
int absoluteValue = Math.abs(-50);  
System.out.println(absoluteValue);       // 50
```

```
System.out.println(Math.min(3, 7) + 2);  // 5
```

- The `Math` methods do not print to the console.
 - Each method produces ("returns") a numeric result.
 - The results are used as expressions (printed, stored, etc.).

Return

- **return:** To send out a value as the result of a method.
 - The opposite of a parameter:
 - Parameters send information *in* from the caller to the method.
 - Return values send information *out* from a method to its caller.
 - A call to the method can be used as part of an expression.



Why return and not print?

- It might seem more useful for the Math methods to print their results rather than returning them. Why don't they?
 - Answer: Returning is more flexible than printing.
 - We can compute several things before printing:

```
double pow1 = Math.pow(3, 4);  
double pow2 = Math.pow(10, 6);  
System.out.println("Powers are " + pow1 + " and " +  
pow2);
```

- We can combine the results of many computations:

```
double k = 13 * Math.pow(3, 4) + 5 - Math.sqrt(17.8);
```

Math questions

- Evaluate the following expressions:
 - `Math.abs(-1.23)`
 - `Math.pow(3, 2)`
 - `Math.pow(10, -2)`
 - `Math.sqrt(121.0) - Math.sqrt(256.0)`
 - `Math.round(Math.PI) + Math.round(Math.E)`
 - `Math.ceil(6.022) + Math.floor(15.9994)`
 - `Math.abs(Math.min(-3, -5))`
- `Math.max` and `Math.min` can be used to bound numbers.
Consider an `int` variable named `age`.
 - What statement would replace negative ages with 0?
 - What statement would cap the maximum age to 40?

Quirks of real numbers

- Some Math methods return double or other non-int types.

```
int x = Math.pow(10, 3);    // ERROR: incompat. types
```

- Some double values print poorly (too many digits).

```
double result = 1.0 / 3.0;  
System.out.println(result);    // 0.3333333333333333
```

- The computer represents doubles in an imprecise way.

```
System.out.println(0.1 + 0.2);
```

– Instead of 0.3, the output is 0.30000000000000004

Type casting

- **type cast:** A conversion from one type to another.
 - To promote an `int` into a `double` to get exact division from `/`
 - To truncate a `double` from a real number to an integer

- Syntax:

(type) expression

Examples:

```
double result = (double) 19 / 5;           // 3.8
int result2 = (int) result;                // 3
int x = (int) Math.pow(10, 3);             // 1000
```

More about type casting

- Type casting has high precedence and only casts the item immediately next to it.

```
- double x = (double) 1 + 1 / 2;           // 1
- double y = 1 + (double) 1 / 2;         // 1.5
```

- You can use parentheses to force evaluation order.

```
- double average = (double) (a + b + c) / 3;
```

- A conversion to `double` can be achieved in other ways.

```
- double average = 1.0 * (a + b + c) / 3;
```



Interactive Programs with Scanner

Input and `System.in`

- **interactive program:** Reads input from the console.
 - While the program runs, it asks the user to type input.
 - The input typed by the user is stored in variables in the code.
 - Can be tricky; users are unpredictable and misbehave.
 - But interactive programs have more interesting behavior.
- **Scanner:** An object that can read input from many sources.
 - Communicates with `System.in` (the opposite of `System.out`)
 - Can also read from files (Ch. 6), web sites, databases, ...

Scanner syntax

- The `Scanner` class is found in the `java.util` package.

```
import java.util.*;    // so you can use Scanner
```

- Constructing a `Scanner` object to read console input:

```
Scanner name = new Scanner(System.in);
```

- Example:

```
Scanner console = new Scanner(System.in);
```

Scanner methods

Method	Description
<code>nextInt()</code>	reads an <code>int</code> from the user and returns it
<code>nextDouble()</code>	reads a <code>double</code> from the user
<code>next()</code>	reads a one-word <code>String</code> from the user
<code>nextLine()</code>	reads a <i>one-line</i> <code>String</code> from the user

- Each method waits until the user presses Enter.
- The value typed by the user is returned.

```
System.out.print("How old are you? "); // prompt
int age = console.nextInt();
System.out.println("You typed " + age);
```

- **prompt:** A message telling the user what input to type.

Scanner example

```
import java.util.*;    // so that I can use Scanner
```

```
public class UserInputExample {  
    public static void main(String[] args) {  
        Scanner console = new Scanner(System.in);
```

```
        → System.out.print("How old are you? ");           age 
```

```
        → int age = console.nextInt();                       years 
```

```
        → int years = 65 - age;  
        System.out.println(years + " years to retirement!");
```

```
    }
```

```
}
```

- Console (user input underlined):

```
How old are you? 29  
36 years until retirement!
```



Scanner example 2

```
import java.util.*;    // so that I can use Scanner

public class ScannerMultiply {
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);

        System.out.print("Please type two numbers: ");
        int num1 = console.nextInt();
        int num2 = console.nextInt();

        int product = num1 * num2;
        System.out.println("The product is " + product);
    }
}
```

- Output (user input underlined):

```
Please type two numbers: 8 6
The product is 48
```

- The Scanner can read multiple values from one line.

Input tokens

- **token:** A unit of user input, as read by the `Scanner`.
 - Tokens are separated by *whitespace* (spaces, tabs, new lines).
 - How many tokens appear on the following line of input?
23 John Smith 42.0 "Hello world" \$2.50 " 19"

- When a token is not the type you ask for, it crashes.

```
System.out.print("What is your age? ");  
int age = console.nextInt();
```

Output:

```
What is your age? Timmy  
java.util.InputMismatchException  
    at java.util.Scanner.next(Unknown Source)  
    at java.util.Scanner.nextInt(Unknown Source)  
    ...
```

Scanners as parameter

- If many methods need to read input, declare a Scanner in main and pass it to the other methods as a parameter (like Graphics)

```
public static void main(String[] args) {
    Scanner console = new Scanner(System.in);
    readSum3(console);
}

// Prompts for 3 numbers and returns their sum.
public static int readSum3(Scanner console) {
    System.out.print("Type 3 numbers: ");
    int num1 = console.nextInt();
    int num2 = console.nextInt();
    int num3 = console.nextInt();
    System.out.println("The sum is " + sum);
}
```