"Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction."
— Jeannette M. Wing

Java's Math class

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

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math.E</td>
<td>2.7182818...</td>
</tr>
<tr>
<td>Math.PI</td>
<td>3.1415926...</td>
</tr>
</tbody>
</table>

static methods in other Classes

- Java includes 8 primitive data types
  - byte, short, int, long, float, double, char, boolean
  - System, String, Graphics, Color, ...

- The Math class contains static methods for common mathematical operations (for which an operator does not exist in Java)
- Call those methods: Math.<MethodName>
  Math.pow(2, 5);

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

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

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

**Clicker 1**

- What is output by the following code?
  ```java
double a = -1.9;
double b = 2.25;
System.out.print( Math.floor(a) + " " + Math.ceil(b) + " " + a);
  ```

A. 3.0
B. -2.0 3.0 -2.0
C. -1.0 3.0 -1.0
D. -1 3 -1.9
E. -2.0 3.0 -1.9
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.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.
  ```java
  int x = Math.pow(10, 3); // ERROR: incompat. types
  ```

- Some double values print poorly (too many digits).
  ```java
  double result = 1.0 / 3.0;
  System.out.println(result);  // 0.3333333333333
  ```

- The computer represents doubles in an imprecise way.
  ```java
  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 for floating point division
  - To truncate a double from a real number to an integer

- Syntax:
  ```java
  (type) expression
  ```

- Examples:
  ```java
  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.
  ```java
  double x = (double) 1 + 1 / 2; // 1.0
  double y = 1 + (double) 1 / 2; // 1.5
  ```

- You can use parentheses to force evaluation order.
  ```java
  double average = (double) (a + b + c) / 3;
  ```

- A conversion to double can be achieved in other ways.
  ```java
  double average = 1.0 * (a + b + c) / 3;
  ```
Returning a value from a method

```
public static type name(parameters) {
    statements;
    ...
    return expression;
}
```

Example:

```
// Returns the slope of the line between the given points.
public static double slope(int x1, int y1, int x2, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    return dy / dx;
}
```

```
slope(5, 11, 1, 3) returns 2.0
```

Common error: Not storing

A return statement DOES NOT send a variable's name back to the calling method.

```
public static void main(String[] args) {
    slope(0, 0, 6, 3);
    System.out.println("The slope is " + result);
    // ERROR: result not defined
}
```

Clicker 2

- Have we (in CS312, before today) used a method that returns a value in lecture?
  A. No
  B. Yes, a few times
  C. Yes, hundreds of time
  D. Lecture?? What lecture?
  E. Maybe?

Return examples

```
// Converts degrees Fahrenheit to Celsius.
public static double fToC(double degreesF) {
    double degreesC = 5.0 / 9.0 * (degreesF - 32);
    return degreesC;
}
```

```
// Computes triangle hypotenuse length given its side lengths.
public static double hypotenuse(int a, int b) {
    double c = Math.sqrt(a * a + b * b);
    return c;
}
```

- You can shorten the examples by returning an expression:

```
public static double fToC(double degreesF) {
    return 5.0 / 9.0 * (degreesF - 32);
}
```
Fixing the common error

- Instead, returning sends the variable's value back.
  - The returned value must be stored into a variable or used in an expression to be useful to the caller.

```java
public static void main(String[] args) {
    double s = slope(0, 0, 6, 3);
    System.out.println("The slope is " + s);
}

public static double slope(int x1, int x2, int y1, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    double result = dy / dx;
    return result;
}
```

Exercise

- In physics, the displacement of a moving body represents its change in position over time while accelerating.
  - Given initial velocity $v_0$ in m/s, acceleration $a$ in m/s$^2$, and elapsed time $t$ in s, the displacement of the body is:
    - Displacement = $v_0 t + \frac{1}{2} a t^2$
- Write a method `displacement` that accepts $v_0$, $a$, and $t$ and computes and returns the change in position.
  - Example: `displacement(3.0, 4.0, 5.0)` returns 65.0
  
  ```java
  public static double displacement(double v0, double a, double t) {
      double d = v0 * t + 0.5 * a * Math.pow(t, 2);
      return d;
  }
  ```

Exercise solution

```java
public static void main(String[] args) {
    double s = slope(0, 0, 6, 3);
    System.out.println("The slope is " + s);
}

public static double slope(int x1, int x2, int y1, int y2) {
    double dy = y2 - y1;
    double dx = x2 - x1;
    double result = dy / dx;
    return result;
}
```

Clicker 3

- What is the output of the following code?
  ```java
  int x = 5;
  int y = 7;
  System.out.print(m(x, y) + " " + x + " " + m(y, x));
  ```

  ```java
  public static int m(int x, int y) {
      x += 2;
      System.out.print(x + " ");
      y -= 2;
      return x * y;
  }
  ```

A. 7 9 35 5 27
B. 7 7 35 7 27
C. 7 5 9 27 35
D. 35 7 5 9 27
E. None of A - D are correct
Exercises

- Write a method to
  - return the int average of 3 ints
  - return the double average of 3 ints
  - return the average of a given number of rolls of 2 six sided dice
  - calculate and return N factorial (N!).
  - return the number of seconds in a given number of years.
  - return the Nth digit of a given integer.
  - return the distance between two points.

Ball solution

// Simulates the dropping of two balls from various heights.
import java.awt.*;
public class Balls {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(600, 600);
        Graphics g = panel.getGraphics();
        int ball1x = 100, ball1y = 0, v01 = 25;
        int ball2x = 200, ball2y = 100, v02 = 15;
        // draw the balls at each time increment
        for (double t = 0; t <= 10.0; t = t + 0.1) {
            g.setColor(Color.GRAY);
            panel.fillRect(0, 0, 600, 600);
            g.setColor(Color.RED);
            double disp1 = displacement(v01, t, 9.81);
            g.fillOval(ball1x, ball1y + (int) disp1, 10, 10);
            double disp2 = displacement(v02, t, 9.81);
            g.fillOval(ball2x, ball2y + (int) disp2, 10, 10);
            panel.sleep(50); // pause for 50 ms
        }
    }
}

Exercise

- If you drop two balls, which will hit the ground first?
  - Ball 1: height of 600m, initial velocity = 25 m/sec downward
  - Ball 2: height of 500m, initial velocity = 15 m/sec downward

- Write a program that determines how long each ball takes to hit the ground (and draws each ball falling).

- Total time is based on the force of gravity on each ball.
  - Acceleration due to gravity $\approx 9.81 \text{ m/s}^2$, downward
  - Displacement $= v_0 t + \frac{1}{2} a t^2$