Model 1 Math Methods

Consider the following methods defined in the `Math` class. (This list isn't exhaustive; the `Math` class has over 90 methods in total!)

Modifier and Type	Method and Description
`static int`	**`abs`**`(int a)` Returns the absolute value of an `int` value.
`static double`	**`log`**`(double a)` Returns the natural logarithm (base *e*) of a `double` value.
`static double`	**`pow`**`(double a, double b)` Returns the value of the first argument raised to the power of the second argument.
`static double`	**`random`**`()` Returns a `double` value with a positive sign, greater than or equal to `0.0` and less than `1.0`.
`static int`	**`subtractExact`**`(int x, int y)` Returns the difference of the arguments, throwing an exception if the result overflows an `int`.

The code for these methods is provided in a source file named *Math.java*. Here is what the definition of the `abs` method looks like:

```
public static int abs(int a) {
    // code omitted
}
```

To use a method from another source file (like *Math.java*), you must first specify the class name:

```
value = abs(-5);       // Error: cannot find symbol
value = Math.abs(-5);  // correct
```

Questions (20 min) Start time:

1. What type of value does `Math.random()` return? Give an example of what a random value might look like.

2. When *defining* a method (like `abs` or `log`), what do you need to specify before the method name and after the method name?

3. Define a method named `average` that takes two integers named `x` and `y` and returns a `double`. Don't write any semicolons or braces.

4. When *using* a method, what do you need to specify before the method name and after the method name?

5. For each method in Model 1, write a Java statement that uses the method and assigns the result to a variable.

*What you wrote for Question #3 is called the method's **signature**. The variables declared inside the parentheses are called **parameters**. When invoking a method, the values you provide are called **arguments**. Since arguments will be assigned to parameters, their types must be compatible.*

6. In the table below, how many parameters and arguments does each method have? What is the relationship between the last two columns?

Method	# Params	# Args
`abs`		
`log`		
`pow`		
`random`		
`subtractExact`		

7. Consider the statement `System.out.println("Price: " + price);` where the value of `price` is 9.99. What is the argument that `println` receives?

8. Consider the statement `System.out.printf("Price: %f", price);` where the value of `price` is 9.99. Why does `println` use *plus* and `printf` use *comma* to specify the arguments?

IMPORTANT: Never use + (string concatenation) with printf. You might accidentally add values to the format string itself, rather than substitute them.