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:

```java
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:

```java
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.

It returns a random `double` value greater than or equal to 0.0 and less than 1.0. For example, the value 0.7851637186510342.

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

Before the name, you need to specify what type of value the method will return. After the name, you need to specify what values are required to use the method.

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.

```
public static double average(int x, int y)
```

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

Before the name, you need to specify the class name (unless the method is in the same class). After the name, you need to specify the required values.

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

```
answer = Math.abs(-5);
length = Math.log(1.2);
amount = Math.pow(3.4, 5.6);
number = Math.random();
result = Math.subtractExact(-78, 90);
```

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`	1	1
`log`	1	1
`pow`	2	2
`random`	0	0
`subtractExact`	2	2

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

```
"Price: 9.99"
```

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?

The `println` method requires only one argument, so plus in this context means concatenate. The `printf` method requires multiple arguments: one for the format string, and others for the values to substitute.

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