Writing Methods

INTRODUCTION & ASSIGNMENT #1

What is a JAVA method?

A method is code that can be executed by "calling" the method.
You have been using the `System.out.println()` method to send output to the console.
The real power of methods comes from the ability to write your own methods to solve problems.

Anatomy of a method

Let us look at a method declaration:

```
public static int randomNumber(int from, int to) { }
```

- A method must first be identified as being either `public` or `private`. A `private` method can only be called from the class that it is in. A `public` method can be called from other classes as well.
- The `static` keyword indicates that this method will not have any information stored in it between calls. For now all our methods will be `static`. A `static` method does a job, and then all of the variables it uses are erased from memory.
- Next comes the RETURN TYPE of the method. If the method is not going to return anything, then the keyword `void` is used for the return type.
- The name of the method comes next. A method name should start with a lowercase letter and it should be descriptive of what the method does.
- All methods must have an open and close parenthesis after the name. There may or may not be anything inside the parenthesis, but they must **always** be there!
- Optionally, there may be a series of variable parameters (also known as arguments) separated by commas. Each parameter consists of a type and a variable name.
- The `{ }` code block is the final ingredient. The actual code that will be executed when the method is called goes between the open and close curly braces!

If a method has a return type other than void, then it must have a return statement that returns the value of a variable or literal of the same type. The method will end after the return statement, but there can be more than one return statement if they are in different branches of a conditional statement.

Without knowing anything about a method, if the declaration is written properly, you should be able to at least make an educated guess as to what it will do, what the parameters mean, and what kind of value, if any, will be returned!

In the example above, can you guess what this method is likely to do?

What do you think will be returned by this method?

Can you say anything specific about the value the method will return?

Writing the method

If you guessed that this method would return a random integer value in the range of `min` to `max`, then you are correct.

Now let's consider the pseudo-code that we could use to fill in the code block for this method:

1. Given variables `min` and `max` as method parameters.
2. Calculate the range of numbers covered by `min` to `max` and store it in variable `range`.
3. Store a random number from 0 to `range` in the variable `rand`.
4. Store the value in `min` plus the value in `rand` in the variable `result`.
5. Return `result`.

Now it is time to turn the pseudo-code into JAVA computer code:

JAVA Code	Pseudo-Code
`public static int randomNumber(int from, int to) {`	Given var `from` and var `to`.
`int range=to-from+1;`	Calculate the range of numbers covered by `from` to `to` and store it in var `range`.
`int rand=(int)(Math.random()*range);`	Store a random number from 0 to `range` in var `rand`.
`int result=rand+min;`	Store the value in `from` + `rand` in var `result`.
`return result;`	Return `result`.
`}`	

Having made our first pass at solving this problem, it is time to test it.
From the `main()` method the `randomNumber()` method can be "called."
In order to test the method, we will put it inside a loop so we can see what it does many times. We will also pick values to use for `from` and `to` that make it easy to see if it is working properly:

Complete JAVA Code:

```
public class RandomNumber {
   public static void main(String[] args) {
      int i=0;
      while (i<10) {
         int r=randomNumber(1,10);
         System.out.println(r);
         i++;
      }
   }
   public static int randomNumber(int from, int to) {
      int range=to-from+1;
      int rand=(int)(Math.random()*range);
      int result=rand+from;
      return result;
   }
}
```

Sample Console Output From Run #

1	2	3	4	5
2	3	9	9	4
2	10	9	5	3
5	2	10	7	9
6	10	4	1	6
2	9	10	10	2
7	10	7	1	1
2	5	4	5	9
8	6	3	8	3
6	9	1	9	7
7	8	6	9	10

To the right of the program you will see a sample of the console output. In order to determine if the method is working as desired, the program was run 5 times. Looking at the results, we can see that the `from` number was generated several times, as was the `to`, and that there are no numbers outside that range.

From the results, can we reasonably determine that the method works as expected?

Will the method still work if we choose a negative number for `from` and for `to`?

What will happen if `to` is less than `from`?

What will happen if `from` and `to` are the same?

For the next assignment, you will enter this program.

Answer all the questions above.

Finally, modify the method so that it works, as one would reasonably expect, under all the circumstances above.

Test your re-written method with each of the following values to the right:

For each pair of values, include sample console output. Identify the `from` and `to` used above the console output!

from	to
1	10
-10	-1
10	5
1	1