Practice Problems: Method Development

1. Tracing Programs

For each program below, show what is displayed on the screen when the code executes.

```
import java.util.Arrays;
public class ReferenceSemantics
{
  public static int mystery(int a, int [] b)
  {
    a++;
    b[0] = 0;
    b = new int[4];
    return a / 2;
  }

  public static void main(String [] args)
  {
    int x = 10;
    int [] y = {3, -3, 3};

    x = x + mystery(x, y);

    System.out.println(x);
    System.out.println(Arrays.toString(y));
  }
}
```

```
public class NestedCalls
{
  public static int mystery1(int a)
  {
    System.out.println("m 1");
    return a / 2;
  }

  public static String mystery2(int b)
  {
    String s = "m 2";
    System.out.println(s);
    for(int i=0; i<b ; i++) {
      s = s + " " + i;
    }
    return s;
  }

  public static void main(String [] args)
  {
    System.out.println("main");
    System.out.println(mystery2(mystery1(4)));
    System.out.println("end main");
  }
}
```

2. Method development walkthrough (StringWalkthrough.java) for calculating the number of letter, say 'i', in a String, and return that number.

a. Start with a method signature: what's the return type, how many arguments does it take, and what is the data type of each argument?

b. Next, decide on an *algorithm*, or recipe, for taking the inputs to this command (the arguments) and computing the outputs (the return value). Write down a step-by-step procedure, in English, for computing the number of 'i's in a String.

c. Next, translate your algorithm into code. Don't forget a return value at the end of the method (and make sure the data type of the return value matches the return type in your method signature).

d. Create a program whose main() method repeatedly reads in Strings from the user and prints out how many 'i's are in that String, until the user enters a word with no 'i'.

e. Modify the method from (c) so that it accepts an argument of type char as well as of type String, and counts how many of that character appear in the String.

f. Have the main() method read in a String and a letter from the user, and print out how many of the letter appear in the String.

3. Another method development walkthrough (PrimeWalkthrough.java) for printing all the prime numbers between 1 and 100.

a. Write a method to compute whether an integer is prime or not. Start with the method signature: what is the return type, how many arguments does it need, and what type are the arguments?

b. Next, decide on an *algorithm*, or recipe, for checking whether a number is prime. Without thinking about the code, write down a simple step-by-step procedure for deciding whether or not a number is prime.

c. Finally, translate your algorithm into code. Remember to return a value (whose type matches the return type in your method signature) at the end.

d. Write the program (including main) that prints out all the prime numbers between 1 and 100. Make calls to the method you defined above.

4. Taste of generalizing Method (Taste.java)

a. Write methods to display a small triangle, a small square, and a small horizontal line in with ASCII art. Make calls to these methods from the main() method so that you create the following figure:

```
XX
XX
================
  ^
 ^^^
^^^^^
XX
XX
```

b. Modify your methods so that they accept a char argument, and the characters that make up each figure match the value of the argument. Then add arguments to your method calls in main() so that the program now displays:

```
XX
XX
----------------
  u
 uuu
uuuuu
OO
OO
```

5. Test of method development.

For the main() method below, come up with a way to break it into multiple methods (also known as "decomposing"), so that the resulting program is better organized. The goal is to print out the entire stairwell with any given number of step, length of run, and height of rise.

```
public class PrintSteps
{
  public static void main(String [] args)
  {
    System.out.println("-----");
    System.out.println("     |");
    System.out.println("     |");
    System.out.println("      -----");
    System.out.println("           |");
    System.out.println("           |");
  }
}
```