Functions Practice

For each of the functions below, write code to implement that function. You may use the FunctionPractice.java file in today's IntelliJ section, or you may start a new project.

1. Write a function that calculates and returns the Euclidean distance between two points in the Cartesian plane. The declaration should be

   ```
   public static double distance(int x1, int y1, int x2, int y2)
   ```

   For instance, `distance(1, 2, 3, -4)` should return the Euclidean distance between (1, 2) and (3, -4), which is approximately 6.3245. Hint: use Math.sqrt().

2. Write a function that sums up a range of numbers, given an upper and lower bound. Both bounds should be included in the sum. The function declaration should be

   ```
   public static int sumRange(int lower, int upper)
   ```

   Example: calling `sumRange(1, 9)` should return 45.

3. Write a function called totalDistance that takes an integer array of ordered pairs, and computes the total distance from the first point to the second, plus the second to the third, etc. The points array will contain (x, y) ordered pairs in this format: {x1, y1, x2, y2, x3, y3, etc}. The function declaration should be:

   ```
   public static double totalDistance(int[] points)
   ```

   Example: calling `totalDistance(new int[] {1, 2, 3, -4, -3, 0})` should add up the distance from (1, 2) to (3, -4), and then the distance from (3, -4) to (-3, 0), which in total is approximately 13.5356.

4. Write a function called sumDigits that takes a single integer argument and returns the sum of its digits (as an integer). For instance, the sum of the digits of the number 324 is 3+2+4=9. The function declaration should be:

   ```
   public static int sumDigits(long num)
   ```

   Note: the long data type in Java can hold bigger numbers than an int can. In Java, an int can hold positive or negative numbers up to approximately two billion (technically between -2^32 and 2^32-1) whereas a long can hold integers up to almost 20 digits (technically between -2^64 and 2^64-1).

   Hint: Use the % operator (remainder) to extract the right-most digit from the number (take the remainder mod 10). Use division by 10 to eliminate that right-most digit entirely and continue until you run out of digits.