Lab 9: Recursion Overview

This lab is designed to illustrate concepts and characteristics of recursion, including the difference between "regular recursion" and "tail recursion," the use of base cases, the concept of the call stack, and arithmetic in return statements. You will write three different public methods and a private helper method in this lab. **You must strictly follow the given method signatures and specifications.**

Specification

You create a `Factorial` class which will implement several versions of a factorial calculation. The recursive definition for factorial is as follows:

Base Case: **`factorial(1) = 1`**
Recurrence Relation: **`factorial(n) = n * factorial(n - 1)`**

The factorial function is undefined for non-positive numbers (including zero); if a non-positive number is passed in, the function should throw an `IllegalArgumentException`.

The `Factorial` class should have the following methods:

```
public static long pureRecursive(int n)
```
A purely recursive function that calculates the factorial of **`n`**. This function should call only itself.

```
public static long tailRecursive(int n)
```
A kickoff method for tail recursion; it should call only the **`tail()`** method (see below).

```
private static long tail(...)
```
A private method called by the tail recursion kickoff method. Students may select the parameters for this method, but it must be present and used. This method should only call itself, and only via tail recursion – i.e., it should call itself on the last line with no other computation after the function call.

```
public static long iterative(int n)
```
An iterative version of the factorial calculation. This method should be an "unwound" version of the **`tailRecursive()`** method outlined above. It should not call itself or any other method but should instead using a looping structure to calculate the factorial.

Submissions

NOTE: Your output must match the example output *exactly*. If it does not, ***you will not receive full credit for your submission***!

Files: Factorial.java
Method: Submit on Canvas

Sample Results (Public Methods)

Param.	Returns
0	*throws IllegalArgumentException*
1	1
2	2
3	6
4	24
5	120
6	720
7	5,040
8	40,320
9	362,880
10	3,628,800
11	39,916,800
12	479,001,600
13	6,227,020,800
14	87,178,291,200
15	1,307,674,368,000