CSC 115 A01 (Java)
Fall 2012 L.Jackson

Assignment #2

Due

Milestone (Submit Part 1): before 3:30 pm on Friday, September 28
Final Submission (Submit Part 2): before 3:30pm on Friday, October 5

<u>Learning Outcomes:</u> Upon successful completion of Assignment #2 Part 1 you will be able to:

- Explain what is a recursive technique to solve a problem
- Use a tracing technique to determine the output of a recursive method
- Write and test Java recursive methods
- Decide when it is better to use iterative or recursive methods

Part 1

Problem statement:

Write three Java static methods, all that calculate x^n for some $n \geq 0$, following the specifications (below), and then answer questions (below) about them. A 'tester' class and main() method is provided.

NOTE: In Part 1 you will only be asked to hand in the answers to the question (No code will be submitted.) However, the methods you write for Part 1 will be submitted with your Part 2 class.

Specifications:

- The first method, called `powerOne()`, must be an iterative method to compute x^n for some $n \geq 0$. (ie, it uses a for loop.)
- The second method, called `powerTwo()`, must be a recursive method (ie, a method that calls itself)to compute x^n by using the following recursive formulation:

  $x^0 = 1$
  $x^n = x * x^{(n-1)}$ if $n>0$
- The third method, called `powerThree()`, must be a recursive method to compute x^n by using the following recursive formulation:

  $x^0 = 1$
  $x^n = (x^{(n/2)})^2$ if $n>0$ and n is even
  $x^n = x * (x^{(n/2)})^2$ if $n>0$ and n is odd

Questions:

- How may multiplications will each of the methods, `powerOne()`,`powerTwo()`, and `powerThree()` perform when computing 3^{19}?
- How may recursive calls will `powerTwo()`, and `powerThree()` perform when computing 3^{19}?

Submitting your Solution:

When complete, submit your answers to the Questions (only!) to the CSc 115 Connex Site using the Tests & Quizzes: Assignment 2 Milestone link before 3:30 on Friday, September 28.

This milestone is a formative exercise: It must be completed and graded but will only be used to give you information about your learning.

Part 2

Problem statement:

Write a Java class called **`MyMath.java`** that can be used to instantiate numerical objects and perform mathematical computations on them. (You will be creating some of the calculations that are available in the Math class!)

Your class needs the following attributes:

x (the base number that is used in the mathematical computations), *n* (an indicator of the number of terms used in the various computations).

Include the following methods:

- A default constructor and one that takes parameters x and n. Do not allow n to be negative.
- Getter and setter methods. Be sure that the setter for n does not allow for negative values.
- An `equals()` method and a `toString()` method.
- an iterative method to compute x^n for some $n \geq 0$. (Similar to: `powerOne()`)
- a recursive method that computes x^n by using the following recursive formulation:

  $x^0 = 1$
  $x^n = x * x^{(n-1)}$ if $n>0$
  (Similar to `powerTwo()`)

- a recursive method that computes x^n by using the following recursive formulation:

  $x^0 = 1$
  $x^n = (x^{(n/2)})^2$ if $n>0$ and n is even
  $x^n = x * (x^{(n/2)})^2$ if $n>0$ and n is odd
  (Similar to `powerThree()`)

- an iterative method to calculate *x!* for some *x≥0*.
  - a recursive method that computes *x!* using the following recursive formulation:

    0!= 1
    *x!= x * (x-1)!* if *x>0*

- an iterative method that uses an *(n+1)* term Taylor series to compute the sine of *x* radians, *sin(x, n)*, using the formula

$$\sin x = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots \quad \text{for all } x$$

- a recursive method that uses an *(n+1)* term Taylor series to compute the sine of *x* radians, *sin(x, n)*, by using the following recursive formulation:

  sin(x,0)= x
  *sin(x,n)= sin(x,n-1) + [(-1)^n * x^(2*n+1)/(2*n+1)!,* if *n>0*

  To avoid using a power function to compute *(-1)^n* use the following compressed if/else*:*

  ```
  // places (-1)^n into variable neg1powerN
  int neg1powerN = (n%2==0 ? 1 : -1);
  ```

  But for the other power calculation and the factorial, you can use a method you wrote previously.
- a second recursive method that uses the Taylor series to computer the sine of *x* radians, *sin(x, n)*. It should use less recursive calls that the one above. An idea: Observe that x – x^3/3! + x^5/5! - . . . . can be factored as follows: x(1-x^2/(3*2) (1+x^2/(5*4) (1- . . . .))).

Create a test program that you can use to test the functionality of your class.

Submitting your Solution:

- When complete submit your **`MyMath.java`** file to the CSc 115 Connex Site using the Assignments: Assignment 2 Submission link before 3:30 on Friday, September 28.