**Prime Numbers Application**

**Author**: John Eric Simmons

**Date Written**: 12/17/2017

**Application stored in my personal GitHub repository at**:

https://github.com/billionayer/CardlyticsEvalCode/tree/master/PrimeNumbers

**Development Environment**: Visual Studio 2015 (Community Edition)

**Development Language**: C#

Application Structure:

1 Project

* **PrimeNumbers** – This project houses the Prime Number calculation application
  + **Program Class** – encapsulates the necessary logic to calculate the prime factors.
    - The core logic to actually factorize a number is in the GetPrimes method.
    - The GetPrimes method implements a crude algorithm that is most likely not the most efficent so I used parallelism in an attempt to make processing more efficient.
    - To make more efficient use of multi-core machines, each number is factorized within a task operation. I use the System.Threading.Task API in the .NET 4.0 framework.
    - The application also peforms a self-check on each prime calculation, by mutiplying all the factors and comparing the result to the corresponding value read from the file. The results of that check is included in the output.

Care was taken to consider situations such as invalid data in the file, passing of null reference objects, and file handling expections should an I/O exception occurred. The application also attempt to handle situations where the files does not exists.

Usage: PrimeNumbers.exe <filePath>

The file shoud have one integer number on each line. The application will disregard any data on a line that does not parse into a ulong.

Cases checked

1. Zero as an input.
2. Negative Values
3. String input
4. Empty lines

