Gregory Asher

The following is a sample output of my program when n = 15.

**Please enter a value for n**

15

**At i = 8; gcd (5,8) = 1 took 4 operations. Completed in: 0.001 milliseconds**

**At i = 9; gcd (5,8) = 1 took 4 operations. Completed in: 0.064 milliseconds**

**At i = 10; gcd (5,8) = 1 took 4 operations. Completed in: 0.102 milliseconds**

**At i = 11; gcd (5,8) = 1 took 4 operations. Completed in: 0.129 milliseconds**

**At i = 12; gcd (5,8) = 1 took 4 operations. Completed in: 0.156 milliseconds**

**At i = 13; gcd (8,13) = 1 took 5 operations. Completed in: 0.183 milliseconds**

**At i = 14; gcd (8,13) = 1 took 5 operations. Completed in: 0.208 milliseconds**

**At i = 15; gcd (8,13) = 1 took 5 operations. Completed in: 0.261 milliseconds**

**Program ended with exit code: 0**

The graph below displays the graphical representation of my modulus operations as a function of N. Additionally I have graphed an upper bound function that describes the worst case scenario.

Below is a graphical representation of the processing time as a function of n operations. In an effort to display the similarity I set the upper limits of the graph to be 1.5 million. The blue graph represents n^2, or the big O of my function. That is, O(n^2)