# **CS315 Homework 3 Report**

**Anthony Snow**

**9/22/16**

**System Specs:**

**OS X El Capitan**

**Version 10.11.14**

**MacBook Pro (13-inch, Mid 2012)**

**Memory: 4 GB 1600 MHz DDR3**

**256 prime numbers generated:**

**0010000101011111001110101011101110110101110101000110111101000010111000110110001100100101011110101001100010100001000011100101101011100010111110110100010001101100110000001010011011010011110101110110011100011000110000001000000111010011010010010011001101000111**

**1101111010110101011010001100101100110101101110101001111100010001111000110110011000110001010010110001111101111010001110110111010000010101110001111000101110110111000001011101101001101000100110000000111010011101101011111011001110111010101001101110010011100101**

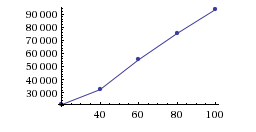
**1001010011001111010110010000100111010001110111000011100011100111100111001111101011110100011111100111100010101111000000010010001110011110000110101110111100101001101011000101110000101101010101010101101111000000101010101001000011110000100001101111010100101001**

**(I did 100 numbers but that would be a lot to repot**

**In number theory, the prime number theorem (PNT) describes the asymptotic distribution of the prime numbers among the positive integers. It formalizes the intuitive idea that primes become less common as they become larger by precisely quantifying the rate at which this occurs. (Wikipedia)**

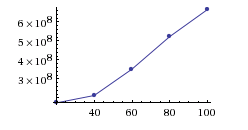
**I feel that my results are because the primes that were created when I did 100 numbers of 256 bits (4) was smaller than that of 16 bit numbers (10).**

**NOTE: The following x-axis on the graphs in the number of numbers generated. And the y-axis is how long the process took**

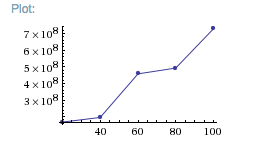
****

**Figure 1: 16 bit Graph**

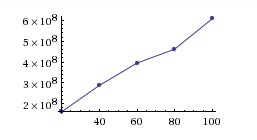
When I generated 100 generated random numbers, 13 of these numbers were prime. The theory states that 1 in (number of bits ) should prime. Which is not the the case. 1 in 7 were prime.

****

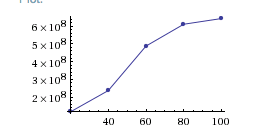
**Figure 2: 32 bit Graph**

****

**Figure 3: 64 bit Graph**

****

**Figure 4: 128 bit Graph**

****

**Figure 5: 256 bit Graph**

**For my test, I would like to conclude that I hypothesize the rate of growth for all functions would be O(n). I say this because the time is strictly related to how many numbers you are generating. There are some outliers, but as seen in class when Dr. Truszczynski performed a sample of running times.**

**The algorithm derived to estimate the running time is Lagrange’s Theorem, so the expected number of iterations is 1/lnx = 1.443/n.**

**Sometimes a computer runs faster then its suppose to. And that’s okay by me.**