

**MAIN TITLE HERE**  
**SUBTITLE HERE**

Created by Aiden Taylor and Noah Pinel

## Contents

<b>1</b>	<b>Table of contents</b>	<b>2</b>
<b>2</b>	<b>Abstract</b>	<b>3</b>
<b>3</b>	<b>Implementation of The Sieve of Eratosthenes in Python</b>	<b>4</b>
3.1	Overview . . . . .	4
3.2	The Implementation . . . . .	4
3.3	Conclusion . . . . .	5
<b>4</b>	<b>Visual Representation of Euler's Phi Function</b>	<b>6</b>
4.1	Overview . . . . .	6
4.2	What is it? . . . . .	6
4.3	The Implementation . . . . .	6
4.4	Conclusion . . . . .	6

**Abstract**

Give a general Overview of our project, explain high level what we each did...

### 3 Implementation of The Sieve of Eratosthenes in Python

#### 3.1 Overview

The Sieve of Eratosthenes is a type of prime sieve. Prime sieves are algorithms that are very good at generating prime numbers up to some upper bound  $n$ , where  $n \in \mathbb{Z}^+$ . The following sections explain the Implementation process of simulating the Sieve of Eratosthenes in the programming language Python.

#### 3.2 The Implementation

My first run in with the sieve of Eratosthenes was during my number theory course, the mechanical nature of the sieve seemed to be a perfect way to utilize a computer to do some interesting math. The entirety of the program relies on the algorithm below, so here are the meat and potatoes of the whole thing,

---

**Algorithm 1** : An algorithm for The Sieve of Eratosthenes

---

**Require:** :  $n > 0$

**Ensure:** : List of prime numbers 2, ...,  $n$

```

1: Sieve( $n$ ) :
2: Populate a boolean list P size 2, ...,  $n$ .
3: Initialize all index values to True.
4: for  $i = 2, \dots, n + 1$ : do
5:   if P[ $i$ ] TRUE
6:     for  $j = i^2, i^2 + 1, i^2 + 3, \dots, n$ : do
7:       set P[ $j$ ] False
8:     end for
9:   end for
10: Return P

```

---

Simply put, we have a list of size 2, ...,  $n$ . We set all values of our list to True, then starting at 2 we loop through each index setting multiples of the respected current position to false. After this has run, we are left with a list where all prime values are unchecked, that is, they remain True while composite numbers are false. Below is a visual of the algorithm working for a list of size 10.

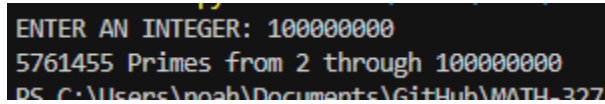
```

ENTER AN INTEGER: 10
0: -->  T, T, T, T, T, T, T, T, T, T
1: -->  T, T, F, T, F, T, F, T, F, F
2: -->  T, T, F, T, F, T, F, F, F, F
3: -->  T, T, F, T, F, T, F, F, F, F
4: -->  T, T, F, T, F, T, F, F, F, F
5: -->  T, T, F, T, F, T, F, F, F, F
6: -->  T, T, F, T, F, T, F, F, F, F
7: -->  T, T, F, T, F, T, F, F, F, F
8: -->  T, T, F, T, F, T, F, F, F, F

```

We can see that after 8 iterations of following the algorithm the only index values set True are 2, 3, 5, 7, i.e., the first 4 primes. Now as cool as it is to visualize the algorithm I feel like finding the

number of primes  $\leq$  some large  $n$  is a lot more interesting. To help with run time I disabled the visual aspect of the algorithm and was able to find a value for the primes up to 100,000,000.



```
ENTER AN INTEGER: 100000000
5761455 Primes from 2 through 100000000
PS C:\Users\nash\Documents\GitHub\MATH_327
```

Using the forbidden site Wolfram Alpha to verify, there are indeed 5,761,455 primes in between 2 and 100,000,000.

### 3.3 Conclusion

The last calculation presented took just about an epsilon under a minute to calculate, which for an algorithm that has been around for thousands of years I feel is very impressive, thus, Eratosthenes sieve is a powerful tool to find primes up to some appropriate positive integer. This problem also displays the interplay between Mathematics and Computer Science and how they complement each other very well. The source code for Sieve.py will be included in our submission with documentation if you're wanting to poke around.

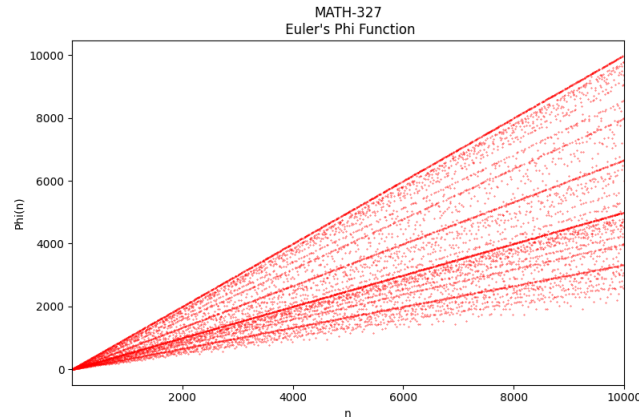


Figure 1: Euler's Phi function plotted up to 10,000

## 4 Visual Representation of Euler's Phi Function

### 4.1 Overview

Euler is one of the most well known Mathematicians ever, the following program I implemented is a visual representation of a function he created, namely the Euler Phi function.

### 4.2 What is it?

Euler's Phi function takes in a positive value  $n$ , and counts the number of relatively prime numbers up to the original  $n$ . For example  $\Phi(10) = |\{1, 3, 7, 9\}| = 4$ , so Euler's function is telling us that from 1-10, there are 4 numbers coprime to 10.

### 4.3 The Implementation

While reading the MATH-327's assigned text book I really enjoyed seeing all the visual representations of the math we were learning. I decided to use python to create a visual of this special function. The process of coding this wasn't too bad, it consisted of reading in an  $n$  value, then for each  $i$  up to  $n$ , I would call the Phi function that I wrote, it would simply just check for a  $\gcd(i, n) == 1$  and if it returned true it was added to a list. After I had this list of all  $1, \dots, n$ 's respected Phi function values I was able to use the python library Matplotlib.pyplot to graph this list against the corresponding  $n$  value, figure(1) is the awesome visual that was generated.

### 4.4 Conclusion

The visual representations that arise from ideas in number theory are amazing to look at, and Euler's Phi function is no excuse. The power computers give us to generate these models is just another example of how technology can give us deep insight into mathematical concepts.