# Eratosthenes Sieve Algorithm for Generating Prime Numbers

Marion Kipsang

July 25, 2023

## 1 Introduction

The Sieve of Eratosthenes is an ancient algorithm for finding all prime numbers up to a given limit. It efficiently identifies prime numbers by iteratively eliminating multiples of each prime found, leaving only the prime numbers at the end of the process.

## 2 Algorithm

The Eratosthenes Sieve algorithm can be outlined as follows:

**Algorithm 1** Eratosthenes Sieve Algorithm

```
procedure SieveOfEratosthenes(limit)
    Let isPrime[0, 1, 2, ..., limit] be a boolean array initialized to true.
    isPrime[0] ← isPrime[1] ← false                 ▷ 0 and 1 are not prime.
    for i ← 2 to √limit do             ▷ Loop through potential prime numbers.
        if isPrime[i] = true then                  ▷ Found a prime number.
            for j ← i² to limit step i do ▷ Mark multiples of i as not prime.
                isPrime[j] ← false
    return all indices i where isPrime[i] = true
```

## 3 Explanation

- The algorithm initializes a boolean array isPrime with all elements set to **true**, representing that all numbers from 0 to the *limit* are initially considered prime candidates.
- We set isPrime[0] and isPrime[1] to **false**, as they are not prime numbers.

- Starting from 2 (the first prime number), the algorithm loops through the array. If a number $i$ is found to be prime (isPrime$[i]$ = **true**), all its multiples from $i^2$ up to the *limit* are marked as not prime (isPrime$[j]$ = **false**).
- After the loop completes, the array will contain **true** for prime numbers and **false** for non-prime numbers.
- The algorithm returns a list of all indices $i$ where isPrime$[i]$ = **true**, which corresponds to the prime numbers up to the specified *limit*.

## 4 Example

Let's apply the Eratosthenes Sieve algorithm to find all prime numbers up to 20.

1. Initialize the array: isPrime = $[\mathbf{true}, \mathbf{true}, \mathbf{true}, \ldots, \mathbf{true}]$.
2. Set isPrime$[0]$ = isPrime$[1]$ = **false**.
3. Start the loop with $i = 2$. Since isPrime$[2]$ = **true**, mark all multiples of 2 as not prime: isPrime$[4]$ = isPrime$[6]$ = isPrime$[8]$ = **false**, and so on.
4. Move to $i = 3$, which is also prime (isPrime$[3]$ = **true**). Mark all multiples of 3 as not prime: isPrime$[6]$ = isPrime$[9]$ = **false**.
5. Continue this process until $i = \sqrt{20}$.
6. At the end, the array isPrime will be: $[\mathbf{true}, \mathbf{true}, \mathbf{true}, \mathbf{false}, \mathbf{true}, \mathbf{false}, \mathbf{true}, \mathbf{false}, \mathbf{false}, \mathbf{false}, \mathbf{true}, \mathbf{fa}$
7. The prime numbers up to 20 are: $[2, 3, 5, 7, 11, 13, 17, 19]$.

## 5 Conclusion

The Sieve of Eratosthenes is a simple yet efficient algorithm for generating prime numbers up to a given limit. Its time complexity is $O(n \log(\log n))$, making it significantly faster than checking each number for primality individually.