10001st Prime

Conrad Warren

June 28, 2024

To quickly calculate all primes less than n, the algorithm "The Sieve of Eratosthenes" is used. The algorithm is as follows:

1. A boolean array `is_prime` of size $n+1$ is created with all elements set to `true`, expect for `is_prime[0]` and `is_prime[1]`, as 0 and 1 are not prime.

2. Iterate over the boolean array, and when an element which marked as `true`, this number is prime and therefor all multiples of this prime are not. So every multiple of this number is marked as `false`, before continuing to iterate over the array `is_prime`.

3. For all values of x, where $x \leq n$: `is_prime[x] = true`, when x is prime and `is_prime[x] = false`, when x is composite.

To find the 10001st prime using the Sieve of Eratosthenes, an upper bound n is needed where $n \geq p_{10001}$. This upper bound can be estimated using the formula:

$$n \approx k \cdot (\log k + \log \log k) \text{ , where } k = 10001$$

Complexity Analysis

The size of the boolean array, `is_prime`, is proportional too $k \cdot \log k$. Where k refers to the kth prime. As k is the input, the overall space complexity is

$$O(n \cdot \log n) \text{ space}$$

Estimating the upper bound of the size of `is_prime` is done in constant time. The time complexity of the Sieve of Eratosthenes algorithm is $O(m \cdot \log \log m)$ where m is the size of the array. As the size of array is proportional to $O(n \cdot \log n)$. The Overall time complexity is:

$$O(n \cdot \log n \cdot \log(\log n + \log \log n)$$