Lab Report 02

Experiment Name: Print prime numbers using the Eratosthenes Algorithm.

Theory:

The Sieve of Eratosthenes is an efficient algorithm for finding all primes smaller than a given number. It works by iteratively marking the multiples of each prime, starting from the multiples of 2.

Code:

```
#include <bits/stdc++.h>
using namespace std;

int main() {

   int prime[1000]= {0};

   int N;
   cout<<"Enter a length: ";
   cin>>N;
   cout<<endl;

   for(int i=2;i<=N;i++){
      prime[i]=i;
   }

   for(int i=2;i<=N;i+=2){
      if(prime[i] % 2 ==0)
         prime[i]=0;
   }

   prime[2]=2;

   for(int i=3;i<=N;i+=3){
```

```
        if(prime[i] % 3==0)
            prime[i]=0;
    }

    prime[3]=3;

    for(int i=5;i<=N;i+=5){
        if(prime[i] % 5 ==0)
            prime[i]=0;
    }

    prime[5]=5;

    for(int i=7;i<=N;i+=7){
        if(prime[i] % 7 ==0)
            prime[i]=0;
    }

    prime[7]=7;

    for(int i=0;i<=N;i++){
            if(prime[i] != 0 )
              cout<<prime[i] <<endl;
    }
}
```

Output:

Result and Discussion:

The code takes an input N from the user, initializes an array prime of size 1000, and then applies the Sieve of Eratosthenes algorithm to find all prime numbers up to the given limit N. Finally, it prints the prime numbers. The code initializes an array prime where each index represents a number. Initially, all values are set to the respective index values.

It then iterates through the array starting from 2, marking multiples of each prime number as non-prime (setting them to 0). After completing the Sieve algorithm, the code prints the remaining non-zero values in the array, which correspond to prime numbers.

Conclusion:

The Sieve of Eratosthenes is a simple and efficient algorithm for finding prime numbers up to a given limit. The code successfully implements the algorithm and provides the prime numbers within the specified range. However, it's important to note that the fixed size of the array (prime[1000]) may limit the range of N for which the algorithm can effectively operate. If a larger range is desired, the array size may need to be dynamically allocated or adjusted accordingly. Additionally, the code could benefit from user input validation to ensure a valid input for N.