CS3610 Project 1

Dr. Jundong Liu

This project is intended as a warm-up assignment to help you refresh your memory of C++ and set up a working environment for future projects.

1 Implemenattion:

A prime number is a number greater than 1 that has no positive integer divisors other than 1 and itself. Conversely, a composite number is any number greater than 1 that is divisible by numbers other than 1 and itself. In other words, a composite number is any number greater than 1 that is not a prime number. To help elucidate the distinction between these two types of numbers, you will write a program that accepts as input a positive integer n and outputs a list of all the prime numbers in the range $[2, n]$ followed by a list of all the composite numbers lying in the same range. To ensure that your program always terminates within a reasonable amount of time, you will separate the prime and composite numbers using the Sieve of Eratosthenes.

The Sieve of Eratosthenes is an algorithm designed to efficiently find all prime numbers less than a given integer limit n. The algorithm indirectly identifies prime numbers by iteratively marking as composite the multiples of each prime found starting from the first prime number 2. The steps of the algorithm are enumerated as follows:

1. Create an ordered list l of all integers in the range $[2, n]$.
2. Initialize the first prime number $p = 2$.
3. Starting from index $2 * p$, iterate over the ordered list l in increments of p and place a mark on each number visited.
4. Find the first unmarked number larger than p in the ordered list l. If no such number is found, exit the algorithm, otherwise, set p to the unmarked value.
5. If $p > \lfloor \sqrt{n} \rfloor$, exit the algorithm, otherwise, return to step 3. $\lfloor ... \rfloor$ is the floor function (`https://en.wikipedia.org/wiki/Floor_and_ceiling_functions`)

An illustrative example of the Sieve of Eratosthenes is provided as follows:

- $n = 10$
  $l = [2, 3, 4, 5, 6, 7, 8, 9, 10]$.
- 1st Iteration:
  $p = 2$ (First prime number)
  Iterate and mark in increments of 2.
  $[2, 3, \cancel{4}, 5, \cancel{6}, 7, \cancel{8}, 9, \cancel{10}]$
- 2nd Iteration:
  p = 3 (Next unmarked number larger than 2)
  Iterate and mark in increments of 3.
  $[2, 3, \cancel{4}, 5, \cancel{6}, 7, \cancel{8}, \cancel{9}, \cancel{10}]$

- 3rd Iteration:
  p = 5
  $5 > \lfloor\sqrt{10}\rfloor$ so exit algorithm.

- Final Result:
  Prime: $[2, 3, 5, 7]$
  Composite: $[\cancel{4}, \cancel{6}, \cancel{8}, \cancel{9}, \cancel{10}]$

Input

A single command line argument containing a positive integer n with the constraints $2 \leq n \leq 30000$

Output

Using space as a delimiter, print all prime numbers less than or equal to n on one line followed by all composite numbers less than or equal to n on a separate line. If the value of n does not lie in the constrained range $[2, 30000]$, output the message `Out of range`. If n is not an integer, output the message `Nan`. If n is not provided as input, output the message `Missing argument`.

Hint

There are many different ways to implement this project. One convenient approach would be using the stoi() function (`http://www.cplusplus.com/reference/string/stoi/`) and "exception handling" mechanism available in C++ (`http://www.cplusplus.com/doc/tutorial/exceptions/`).

Sample Test Cases

Test Case 1

```
$ ./a.out 10
2 3 5 7
4 6 8 9 10
```

Test Case 2

```
$ ./a.out
Missing argument
```

Turn In

Submit your source code through blackboard. If you have multiple files, package them into a zip file.

Grading

Total: 100 pts.

- **10**/100 - Code style, commenting, general readability.
- **10**/100 - Compiles.
- **5**/100 - Follows provided input and output format.
- **75**/100 - Successful implementation of the Sieve of Eratosthenes.