

# 1 Description of the used algorithm

Sieve of Eratosthenes is used to find all prime numbers below certain limit  
Let's call this limit  $n$

It starts with number 2, which is the first prime number and marks all multiples of this number (up to predefined limit) as composite (not prime), those numbers will be later ignored

Then it takes next available number (3) and does the same thing

This is repeated until there are no more numbers below the limit which are neither prime nor crossed out

Then we return the list of all non-crossed out (prime) numbers

We used more optimized version of this algorithm and cross out composites only up to  $\sqrt{n}$  in the main loop

# 2 Functional description of the application

First we define the limit, we name this limit as *num* which will decide how many numbers we will check, either by user interface or we hard code it in  
We define boolean list which will be used to distinguish between prime and composite numbers

We start with number 2 and assign it to variable named  $p$

Then we calculate the primes using Sieve of Eratosthenes using nested while loops

External loop goes through numbers smaller than  $\sqrt{num}$ , starting with current value of  $p$

It checks if the number we are currently was checked out by checking the value of boolean table

if it was not checked out it gets a new number which is the  $p$  multiplied by 2, then it goes into inner loop

inner loop sets all multiplicities of  $p$  as crossed out by setting their value in boolean table to false

then we increment the  $p$  and the whole loop repeats until we run out of numbers

we return array of prime numbers to function which prints those numbers

## **2.1 Input data format**

There is single input, variable named *num* which is the upper limit of numbers to checked

It is a simple int variable, it cannot be less than 2 and has to be a whole number.

## **2.2 Output text on console**

As an output we put out all input prompts which ask for an upper limit, and the list of prime numbers found by our algorithm.

## **2.3 Format of output data**

Output data is a string

# **3 Description of designed code structure**

## **3.1 Own implementation tests**

Comments

## **3.2 Reference tests**

Source of reference values

Comparision with our implementation

Comments

# **4 Tests**