Introduction to Python

Outline for Sieve of Aristophanes

Here is one possible approach. It has some glaring inefficiencies, but we can deal with them later.

The list “x” will represent the list containing the range of numbers to check for primes. The list “primes”, will have all the primes that we find. We initialize the primes list with 2, which is the first prime.

The basic idea is that each time we find a prime \( p \) in the list \( x \), we set \( x[p]=0 \) and all of its multiples as well.

We then locate the next prime in list \( x \). It will be at the very next position \( k \), such that \( x[k]!=0 \). We continue this process till we are done.

How do we know when we are finished? When all the elements of \( x \) have been set to 0.

1. Basic setup.

\[
n = \text{int(input('What is the range? '))}
\]

\[
x = []
\]

\[
\text{primes} = [2]
\]

# Fill the array \( x \) with the numbers 1 – \( n \) so that \( x[i]=i \).

\[
p = 2 \quad \# \text{p is set =2, our first prime}
\]

\[
x[0] = x[1] = x[2] = 0 \quad \# \text{manually set the first 3 positions of x to 0}
\]

2. While we are not yet done: # list \( x \) still has some non-zero elements

- set \( x =0 \) at all positions that are multiples of the current prime \( p \)
- search for the next prime. Start one position to the right of the last one that you found (i.e. start looking at position \( p+1 \))
- when you find the prime, call it \( p \).
  - append \( p \) to primes
  - set its position to 0
  - go back to step 2

3. We are done, so print a “table” of the primes, 5 across.