|  |  |
| --- | --- |
|  | **Sieve of Eratosthenes** |
|  | Given a number n, print all primes smaller than or equal to n. It is also given that n is a small number.  **Example:**  ***Input :****n =10* ***Output :****2 3 5 7*  ***Input :****n = 20* ***Output:****2 3 5 7 11 13 17 19* |
|  | The sieve of Eratosthenes is one of the most efficient ways to find all primes smaller than n when n is smaller than 10 million or so .  Following is the algorithm to find all the prime numbers less than or equal to a given integer n by the Eratosthene’s method:  When the algorithm terminates, all the numbers in the list that are not marked are prime.  **Explanation with Example:**  Let us take an example when n = 50. So we need to print all prime numbers smaller than or equal to 50.  We create a list of all numbers from 2 to 50. |
|  | Lightbox |
|  | **def** SieveOfEratosthenes(n):        # Create a boolean array      # "prime[0..n]" and initialize      #  all entries it as true.      # A value in prime[i] will      # finally be false if i is      # Not a prime, else true.      prime **=** [True **for** i **in** range(n**+**1)]      p **=** 2  **while** (p **\*** p <**=** n):            # If prime[p] is not          # changed, then it is a prime  **if** (prime[p] **==** True):                # Update all multiples of p  **for** i **in** range(p **\*** p, n**+**1, p):                  prime[i] **=** False          p **+=** 1        # Print all prime numbers |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |