

## Problem 7

### Find the 10001<sup>st</sup> prime

We do not know what answer to expect so we will try to solve this problem using trial division. However, if a good upper bound for the target prime is known in advance, using a sieve of Eratosthenes is a much more efficient method.

Some useful facts:

1 is not a prime.

All primes except 2 are odd.

All primes greater than 3 can be written in the form  $6k \pm 1$ .

Any number  $n$  can have only one primefactor greater than  $\sqrt{n}$ .

The consequence for primality testing of a number  $n$  is: if we cannot find a number  $f$  less than or equal  $\sqrt{n}$  that divides  $n$  then  $n$  is prime: the only primefactor of  $n$  is  $n$  itself

Let's design an algorithm that tests the primality of a number  $n$  based on these facts:

```
Function isPrime(n)
if n=1 then return false
else
if n<4 then return true  //2 and 3 are prime
else
if n mod 2=0 then return false
else
if n<9 then return true    //we have already excluded 4,6 and 8.
else
if n mod 3=0 then return false
else
    r=floor( $\sqrt{n}$ )  // $\sqrt{n}$  rounded to the greatest integer r so that r*r<=n
    f=5
    while f<=r
        if n mod f=0 then return false (and step out of the function)
        if n mod (f+2)=0 then return false (and step out of the function)
        f=f+6
    endwhile
    return true (in all other cases)
End Function
```

We can use this function with:

```
limit=10001
count=1 //we know that 2 is prime
candidate=1
repeat
    candidate=candidate+2
    if isPrime(candidate) then count=count+1
until count=limit
output candidate
```