# Euler 0007

## The Problem:

Prime numbers are numbers that are only divisible by themselves and 1. Let's generate primes!
**What is the 1001st prime?**

## The Approach:

We can re-use some of the prime factorization code, and just check the stacked prime factors that we have accumulated.

We start with the primes 2 and 3 and just move up numbers up through numbers by 6 and check each neighbor. This mathematically works out because of basic rules of sieving.

It is definitely not an optimal algorithm, but should be good enough for 1000 primes. So I'm happy with that.

## The Code:

```
prime_count = 10001
prime_factors = [2,3] #we can just start with 2 and 3, because we already know them

#simple prime checker
def add_if_prime(number, primes):
    is_prime = True
    #check through all the primes we have gathered so far.
    for prime in primes:
        if number % prime == 0:
            is_prime = False
            break

    if is_prime:
```

```
        primes += [number]

    return primes

#actual loop to iterate through
current = 6
while len(prime_factors) < prime_count:
    add_if_prime(current-1, prime_factors) #check the number before
    add_if_prime(current+1, prime_factors) #check the number after
    current += 6 #loop through incrementing by 6

print(prime_factors[prime_count-1])
```

Revision #2
Created 2025-07-08 14:09:36 UTC by Maxwell
Updated 2025-07-17 14:51:31 UTC by Maxwell