Euler 0026

The Problem:

Find the value of a,b of n^2 + an + b where |a| < 1000 and |b| <= 1000 where starting from n = 0 and incrementing, there is the largest consecutive chain of primes.

Considerations and Approach:

The way that we can approach this is by incrementing a from -999 to 999 and b from -1000 to 1000.

We only have to generate 1000^2 + 1000*1000 + 1000 as the upper limit of primes that we need to search as well.

We are going to generate all of these primes and as we increment we will first check that the sum of a + b > 1 if it isn't then we are not starting off with a prime. Once we have done this initial check to cull all of the numbers that for sure won't work, then we can check for primality (we could just start the incrementers at the best spot but ehhh...)

To check for primality we generate a list of primes up to the upper limit that we are searching and then convert this list into a dictionary that returns true on a hit. This way, we aren't searching through a list, we are just doing a direct dictionary lookup and verifying that it is indeed prime.

(Without setting up this dictionary lookup this program took minutes and it only got through a = -800 because searching a rather large list using the *in* operator gets expensive near the end of the list. Doing a dictionary only made this whole operation about 1 second, after the prime generation)

Not my most brilliant solution, but fun nonetheless and still efficient enough for the problem space.

The Code:

```
#This generates primes up to a threshold using sieving.
#I've used this sieve generator in many of my problems.
prime_threshold = 2*(1000**2) + 1000
number_list = list(range(2, prime_threshold+1))
prime_list = []
```

```
current = 0
total_iteration = 0
while current < len(number_list):
    if number_list[current] != -1:
        prime = number_list[current]
        prime_list += [prime]

        increment = current + prime
        while increment < len(number_list):
            number_list[increment] = -1
            increment += prime
            total_iteration += 1

        #print(number_list)
        #print(prime_list)
    current += 1

######

#convert the list into a dictionary of {int:True}
#so that .get will return hits or null/false (I don't remember what it does)
quick_prime_list = {}
for prime in prime_list:
    quick_prime_list[prime] = True

#######

#Set up the variables
a = -999
b = -1000
longest_n = [a,b]
longest_n_length = 0
upper_stop_a = 1000
upper_stop_b = 1000 #b is inclusive

#increment a to the upper stop
while a < upper_stop_a:
    #increment b to the upper stop inclusive
```

```
    while b <= upper_stop_b:
        n = 0
        #if it is less than 2 then the first number isn't prime...
        if a + b > 1:
            #now we just keep incrementing n while checking the prime list
            while quick_prime_list.get(n**2 + a*n + b):
                n += 1

            #is this the highest n that we have generated???
            if n > longest_n_length:
                longest_n_length = n
                longest_n = [a,b]
        b += 1

    #reset the a loop
    b = -1000
    a += 1

    #use as debug to check the status of the a (main) loop
    #if a % 10 == 0:
    #    print(a)
```

Revision #2
Created 2026-01-26 21:39:06 UTC by Maxwell
Updated 2026-01-26 21:49:31 UTC by Maxwell