Solution sequences

July 25, 2021

^,1, 2, 4, 14, 34, 5154, ?????, ... (^ = cannot be extended to the left). An "opposite" sequence related to this one: ^,1, 2, 4, 14, 6, ?, 3, 427010, The "middle" sequence: ^,1, 1|2, 1|2|4, 1|2|4|14, 1|2|4|6|14|34, ? The evenness of the numbers isn't a useful pattern as far as I know. HINT: Euclid's Proof of the Infinitude of Primes [https://primes.utm.edu/notes/proofs/infinite/euclids.html]

The key functions are product plus one and prime-order. The numbers in sequences are the orders of the primes. All three sequences are made by product-plus-one operating on a set(no duplicate primes). In the first sequence(as listed here) all new primes are added. In the next, the greatest and in the last, the smallest prime factor.

Using SageMath 8.6: [https://www.sagemath.org/]

```
In [66]: print(seqVerticalBar(seq3(2,8)))
```

```
1, 1|2, 1|2|4, 1|2|4|14, 1|2|4|6|14|34, 1|2|4|6|14|34|234477, 1|2|4|6|14|34|101|111|174|3345|23
```

```
In [67]: print(seqPrint(seq1(2,8)))
```

```
1, 2, 4, 14, 34, 5154, 29266, 86161548656
```

```
In [68]: print(seqPrint(seq2(2,8)))
```

```
1, 2, 4, 14, 6, 16, 3, 427010
```

1 Functions

```
In [69]: #Greatest prime factor of a number
         def gpf(n):
             return last(factor1(n))

         #Least prime factor of a number
         def lpf(n):
             return first(factor1(n))
```

```
#Product of a set plus one
def prodPlusOne(L):
    L=set(L)
    res=1
    for e in L:
        res = res*e
    return res+1

#First element of a list.
def first(L):
    L=list(L)
    return(L[0])

#Last element of a list.
def last(L):
    return(list(L)[-1])

#Unique factors of a number(no dupliates). Or factors as a set.
#https://ask.sagemath.org/question/33493/list-of-prime-factors-with-repetition/
def factor1(n):
    F = factor(n)
    L=[p for (p, m) in F for _ in range(m)]
    L=list(set(L))
    L.sort()
    return L

#Index of prime 2=>1 3=>2 ...
def indexOf(p):
    return prime_pi(p)

#Change whole list of primes to indexes.
def indexOfL(L):
    return map(indexOf,L)

#Change whole list-of-lists of primes to indexes.
def indexOfLL(LL):
    LL2=[]
    for L in LL:
        LL2.append(map(indexOf,L))
    return LL2

#gpf are added.
def seq1(p,n):
    L=[p] #Starting prime
    for i in range(1,n):
        L.append(gpf(prodPlusOne(L)))
    return indexOfL(L)
```

```
#lpf are added.
def seq2(p,n):
    L=[p] #Starting prime
    for i in range(1,n):
        L.append(lpf(prodPlusOne(L)))
    return indexOfL(L)

#All new factors are added(the set is represented sorted).
def seq3(p,n):
    L=[p]
    LL=[L]
    for i in range(1,n):
        L=list(set(L+(factor1(prodPlusOne(L)))))
        L.sort()
        LL.append(L)
    return indexOfLL(LL)

#Regular seq print OEIS format.
#https://en.wikipedia.org/wiki/On-Line_Encyclopedia_of_Integer_Sequences
def seqPrint(L):
    return ', '.join(map(str,L))

#Print with Vertical bar.
#https://en.wikipedia.org/wiki/Vertical_bar
def seqVerticalBar(LL):
    LL2=[]
    for L in LL:
        LL2.append('|'.join(map(str,L)))
    return ', '.join(map(str,LL2))
```
