Calculating the Smarandache Numbers

Jon Perry
51 High Street
Belbroughton
West Midlands
DY9 9ST
England

Abstract
The Smarandache Numbers are:
1,2,3,4,5,3,7,4,6,5,11,4,13,7,5,6,17,6,19,5,7,11,23,4,10,13,9,7,29,5,31,8,11,17,7,6,37, 19,13,5,41,7,43,11,6,23,47,6,14,10,17,13,53,9,11,7,19,29,59,5,61,31,7,8,13,11,67,17, 23,7,71, 6,73,37,10,19,11,13,79,6,9,41,83,7, ...
and defined as the smallest integer m such that n divides m! Finding the exact value of a(n) is an open problem, and this paper presents an effective algorithm for determining the value of a(n).

Keywords
Smarandache functions, factorial, prime numbers

Introduction
The process involved is fairly simple, and we need to know the factorisation of n. From this factorisation, it is possible to exactly calculate by which m each prime is satisfied, i.e. the correct number of exponents appears for the first time. The largest of these values gives a(n).

Satisfying p^k
To satisfy p^k, we find the lowest m such that p^k divides m!.

For example, if we look at $3^4=81$, then m=9 suffices and is also the lowest possible value of m we can achieve.

We can see that m=9 suffices, as 9!=1.2.3.4.5.6.7.8.9, of which 3,6 and 9 are multiples of 3, and 9 happens to be 3^2. As 3, 6 and 9 are the first multiples of 3, this implies m=9 is minimal.

The key to finding m lies in the value of k, and with the distribution of 3's over the integers.

The pattern of divisibility by 3, beginning with 1, is;

0 0 1 0 0 1 0 0 2 0 0 1 0 0 1 0 0 2 0 0 1 0 0 1 0 0 3 0 ....

For the purpose of the Smarandache numbers, we can remove the 0's from this, as we are only concerned with accumulating enough 3's.

(A) 1 1 2 1 1 2 1 1 3 1 1 2 1 1 2 1 1 3 1 1 2 1 1 2 1 1 4 1 ....

The pattern present here can be generalized at a basic level to allow us to calculate the values of the sums whenever a number appears for the first time.

This gives us the sub-sequences 1, 112, 112112113, etc…, and we are interested in the sums of these, i.e.:

(B) 1, 4, 13, 40 ...

This is the partial sums of 1+3+9+27+…, and this is result of evaluating $(3^n-1)/2$.

Now we can deduce the value of m from k, where does k appear in B? Our k in the example was 4, and this appears as B(2). This means that to reach 3^4 we need 3 terms from A $(=3^{(2-1)})$, and multiplying by 3 gives the answer we require of 9.

But how about 3^{333}? To calculate m for this, we reduce in by as many possible of the terms of A.

A fuller list of A is:

(pari/gp code)
```
three(n)=(3^n-1)/2
for (n=1,8,print1(three(n)",")))
```

1,4,13,40,121,364,1093,3280,

364 is too large, but 121 is Ok. 333-121=212, and again 212-121=91.

121 is A(5), so the data collected so far is [2*5]

Continuing, 91-2*40=11, and 11-2*4=3, and 3=1*3, thus we have the data [2*5, 2*4, 2*2, 3*1].

To interpret this data, we just re-apply it to the distribution of 3's. 2*5 means that we need $2*3^4$ consecutive multiples of 3 – by this stage we have satisfied 3^{242}. 2*4 means that we add a further $2*3^3$ multiples of 3, 2*2 means that we add a further $2*3^1$ multiples of 3, and finally we add 3*1 multiples of 3.

The whole sum is therefore 2*81+2*27+2*3+3*1=162+54+6+3=225, and this gives us the answer directly: (225*3)! = 675! is the smallest factorial that 3^{333} divides.

This can be proven with a small Pari program:

```
? for(i=1,2000,if(i!%3^333==0,print1(i);break))
675
```

Calculating a(n)

Then we need to calculate the m value for each prime and exponent, and a(n) is the largest.

This Pari/GP code performs the necessary calculations

```
{
findm(x,y)=local(m,n,x1);
m=0;n=1;x1=x-1;
while (((x^n-1)/x1)<=y,n++);n--;
while (y>0,
while (((x^n-1)/x1)<=y,y-=((x^n-1)/x1);m+=(x^(n-1)));n--);
x*m
}
```

This is the findm() function. n is boosted until larger than necessary, and then trimmed down one so that is must be less than or equal to y. Then y is decreased by the largest possible value of (x^n-1)/(x-1) possible until y=0. m is continually incremented throughout this process as appropriate, and the returned value is x*m.

```
{
smarandache(n)=local(f,fl,ms);
if (n==1,1,
f=factor(n);fl=length(f[,1]);
ms=vector(fl,i,0);
for (i=1,fl,ms[i]=findm(f[i,1],f[i,2]));
vecmax(ms))
}
```

The smarandache() function returns 1 if n is 1, otherwise it creates the ms vector of lowest possible m values, and returns the largest value.

The program results in this data:

```
?for (i=1,100,print1(smarandache(i)","))
1,2,3,4,5,3,7,4,6,5,11,4,13,7,5,6,17,6,19,5,7,11,23,4,10,13,9,7,29,5,31,8,11,17,
7,6,37,19,13,5,41,7,43,11,6,23,47,6,14,10,17,13,53,9,11,7,19,29,59,5,61,31,7,8,1
3,11,67,17,23,7,71,6,73,37,10,19,11,13,79,6,9,41,83,7,17,43,29,11,89,6,13,23,31,
47,19,8,97,14,11,10,
```

which give a 100% correlation with the sequence given in the abstract.

At 100Mhz, it takes about 1 minute to generate the sequence to n=10000.

Reference:

Neil Sloane, The Encyclopaedia of Integer Sequences, Sequence # A002034,
http://www.research.att.com/cgi-
bin/access.cgi/as/njas/sequences/eisA.cgi?Anum=A002034