CALCULATING THE SMARANDACHE FUNCTION FOR POWERS OF A PRIME

J. R. SUTTON
(16a Overland Road. Mumbles, SWANSEA SA3 4LP, UK)

Introduction
The Smarandache function is an integer function, S, of an integer variable, n. S is the smallest integer such that S! is divisible by n. If the prime factorisation of n is known

\[ n = \prod m_i p_i \]

where the \( p_i \) are primes then it has been shown that

\[ S(n) = \text{Max} \left( \left( S(m_i p_i) \right) \right) \]

so a method of calculating S for prime powers will be useful in calculating S(n).

The inverse function
It is easier to start with the inverse problem. For a given prime, \( p \), and a given value of S, a multiple of \( p \), what is the maximum power, \( m \), of \( p \) which is a divisor of \( S! \)? If we consider the case \( p=2 \) then all even numbers in the factorial contribute a factor of 2, all multiples of 4 contribute another, all multiples of 8 yet another and so on.

\[ m = \text{S DIV}2 + (\text{S DIV}2)\text{DIV}2 + ((\text{S DIV}2)\text{DIV}2)\text{DIV}2 + \ldots \]

In the general case

\[ m = \text{S DIV}p + (\text{S DIV}p)\text{DIV}p + ((\text{S DIV}p)\text{DIV}p)\text{DIV}p + \ldots \]

The series terminates by reaching a term equal to zero. The Pascal program at the end of this paper contains a function invSpp to calculate this function.
Using the inverse function

If we now look at the values of S for successive powers of a prime, say p=3,

\[
\begin{array}{cccccccccccc}
m & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & \ldots \\
S(3^m) & 3 & 6 & 9 & 9 & 12 & 15 & 18 & 18 & 21 & 24 & \ldots \\
\end{array}
\]

where the asterisked values of \(m\) are those found by the inverse function, we can see that these latter determine the points after which \(S\) increases by \(p\). In the Pascal program the procedure `tabsmarpp` fills an array with the values of \(S\) for successive powers of a prime.

The Pascal program

The program tests the procedure by accepting a prime input from the keyboard, calculating \(S\) for the first 1000 powers, reporting the time for this calculation and entering an endless loop of accepting a power value and reporting the corresponding \(S\) value as stored in the array.

The program was developed and tested with Acornsoft ISO-Pascal on a BBC Master. The function `time` is an extension to standard Pascal which delivers the timelapse since last reset in centi-seconds. On a computer with a 65C12 processor running at 2 MHz the 1000 \(S\) values are calculated in about 11 seconds, the exact time is slightly larger for small values of the prime.

```pascal
program TestabSpp(input,output);
var t,p,x: integer;
Smarpp:array[1..1000] of integer;

function invSpp(prime,smar:integer):integer;
var m,x:integer;
begin
  m:=0;
x:=smar;
repeat
  x:=x div prime;
m:=m+x;
until x<prime;
invSpp:=m;
end; {invSpp}
```
procedure tabsmarpp(prime,tabsize:integer);
var i,s,is:integer;
exit:booleani
begin
exit:=false;
i:=1;
is:=1;
s:=prime;
repeat
repeat
Smarpp[i]:=s;
i:=i+1;
if i>tabsize then exit:=true;
until (i>is) or exit;
s:=s+prime;
is:=invSpp(prime,s);
until exit;
end; {tabsmarpp}

begin
read(p);
t:=time;
tabsmarpp(p,1000);
writeln((time-t)/100);
repeat
read(x);
writeln('Smarandache for ',p,' to power ',x,' is ',Smarpp[x]);
until false;
end. {testabsp}