Problem 21

Evaluate the sum of all amicable pairs under 10000

Let d(n) be defined as the sum of proper divisors of n (numbers less than n which divide evenly into n).
If d(a) = b and d(b) = a, where a ≠ b, then a and b are an amicable pair and each of a and b are called amicable numbers.

For example, the proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110; therefore d(220) = 284. The proper divisors of 284 are 1, 2, 4, 71 and 142; so d(284) = 220.

Evaluate the sum of all the amicable numbers under 10000.

There is a possible ambiguity here: should both members of a pair be below 10000 or does the condition also include pairs (a,b) for which a<10000, but b not.
If you displayed all pairs you found, you will have noticed that there is no such pair, so we will not bother.

A second issue is: there exist so-called perfect numbers: numbers so that d(a)=a.
Well the problem statement says that is not an amicable pair.

First suppose we have a function SumOfProperDivisors(n) how can we use this efficiently?
Let us concentrate on that first.

Suppose one has coded:

```
for a=2 to 9999
for b=2 to 9999
  if a<>b then
  if SumOfProperDivisors(a)=b and if SumOfProperDivisors(b)=a then....
```

one invokes the function if SumOfProperDivisors 9998*9997=99,950,006 times
If one replaces this with

```
for a=1 to 9999
for b=a+1 to 9999
   if SumOfProperDivisors(a)=b and if SumOfProperDivisors(b)=a then....
```

one still invokes this function 9998*9997/2 times.
That can be done much more effiently: if we calculate b=SumOfProperDivisors(a) then SumOfProperDivisors(b) should be equal to a.
So our main program now becomes:

```
sum=0
for a=2 to 9999
  b= SumOfProperDivisors(a)
  if b>a then
  if SumOfProperDivisors(b)=a then
      sum=sum+a+b
output sum.
```

Question: what does that "`if b>a then` " there?
Even if for all a SumOfProperDivisors(a) were greater than a we would invoke our function only 9998*2 times, an almost 5000 times speed increase!

Let us now concentrate on the function SumOfProperDivisors.
Here is a very naive version we will try to improve on.

```
Function SumOfProperDivisors(n)
sum=1
for f=2 to n-1
   if n mod f=0 then sum=sum+f
return sum
EndFunction
```

Now look what happens when we list the proper divisors of say 36:
Here they are:
1
2 18
3 12
4 9
6 (6)
except for 1 they come in nice pairs so that their product is 36. When we found one of the two in fact we know the other. Special care is needed because 36 is a perfect square: 6 is found twice. To find all pairs we need only to check up to $\sqrt{n}$!
A second improvement comes from the realisation that odd numbers cannot have even numbers as divisors. (The converse is not true: even numbers can have odd divisors).

Second, improved version:

```
Function SumOfProperDivisors(n)
If n=1 then return 0 else
r=floor(√n)
//first take into account the case that n is a perfect square.
if r*r=n then sum=1+r r=r-1 else sum=1
if odd(n) then f=3 step=2 else f=2 step=1
while f<=r
    if n mod f=0 then sum=sum+f+n div f
    f=f+step
return sum
EndFunction
```

In fact this can be even improved on using the prime factorisation of n.
If we know the prime factorisation if n it is easy to calculate the sum of the divisors of n from that.
The sum of the proper divisors of n is then the sum of the divisors of n minus n itself.
More on this can be found on the companion website to Project Euler:
http://mathschallenge.net/index.php?section=faq&ref=number/sum_of_divisors
(clickable link).

This can be coded as: (see next page)

```
Function SumOfDivisors(n)
sum=1
p=2
while p*p<=n and n>1
    if n mod p=0 then
         j=p*p
         n=n div p
         while n mod p=0
               j=j*p
               n=n div p
          sum=sum*(j-1)
          sum=sum div (p-1)
    if p=2 then p=3 else p=p+2
if n>1 then sum=sum*(n+1)
return sum
EndFunction

Function SumOfProperDivisors(n)
return SumOfDivisors(n)-n
EndFunction
```

Notes:
The line:
`while p*p<=n and n>1`
prevents us from checking prime factors greater then $\sqrt{n}$.
Note that as more and more primefactors get divided out, n gets smaller and smaller and the upperlimit for p decreases accordingly!
The line:
`if n>1 then sum=sum*(n+1)`
covers the case that one prime factor greater than $\sqrt{n}$ remains.