Perfect numbers

Various old cultures searched a connection between a number and the sum of its positive divisors. Numbers with special properties were often given a mystical interpretation. Positive integers that are equal to the sum of their positive divisors are called *perfect numbers*. Here, 1 is seen as a positive denominator of every number, and a number is not seen as a positive divisor of itself. Like that, the ancient Greeks already knew the first four positive divisors (see the table underneath), and currently there are 44 positive divisors.

positive divisor	sum of divisors
6	$1 + 2 + 3$
28	$1 + 2 + 4 + 7 + 14$
496	$1 + 2 + 4 + 8 + 16 + 31 + 62 + 124 + 248$
8128	$1 + 2 + 4 + 8 + 16 + 32 + 64 + 127 + 254 + 508 + 1016 + 2032 + 4064$

Integers of which the sum of the positive divisors is smaller than the number itself are called *deficient numbers*, and integers of which the sum of their positive divisors is larger than the number itself are called *abundant numbers.*

Assignment

1. Write a function sum_divisors, that prints the sum of the positive divisors of a given integer (that is given to the function as a parameter). This way, for the value 9, the function should print the result 4 (= $1 + 3$), and the value 10 results in the sum 8 (= $1 + 2 + 5$).

   def sum_divisors(n)

2. Use the function sum_divisors to write a function kindofnumber, that prints a string for a given integer that indicates the kind of number: perfect, deficient or abundant. That way, the function should print perfect for the value 28.

   def kindofnumber(n)

Example

```
>>> sum_divisors(28)
28
>>> kindofnumber(28)
'perfect'
>>> sum_divisors(29)
1
>>> kindofnumber(29)
'deficient'
>>> sum_divisors(30)
42
>>> kindofnumber(30)
'abundant'
```

Verschillende oude culturen hielden zich bezig met het zoeken naar verbanden tussen een getal en de som van zijn echte delers. Aan getallen met speciale eigenschappen werd dan ook vaak

een mystieke interpretatie gegeven. Zo worden positieve gehele getallen die gelijk zijn aan de som van hun echte delers *perfecte getallen* genoemd. Hierbij wordt 1 als een echte deler van elk getal beschouwd, en wordt een getal niet als een echte deler van zichzelf aanzien. Zo kenden de oude Grieken reeds de eerste vier perfecte getallen (zie onderstaande tabel), en zijn er momenteel 44 perfecte getallen gekend.

perfect getal	**som van delers**
6	$1 + 2 + 3$
28	$1 + 2 + 4 + 7 + 14$
496	$1 + 2 + 4 + 8 + 16 + 31 + 62 + 124 + 248$
8128	$1 + 2 + 4 + 8 + 16 + 32 + 64 + 127 + 254 + 508 + 1016 + 2032 + 4064$

Positieve gehele getallen waarvan de som van de echte delers kleiner is dan het getal zelf worden *gebrekkige getallen* genoemd, en positieve gehele getallen waarvan de som van de echte delers groter is dan het getal zelf worden *overvloedige getallen* genoemd.

Opgave

1. Schrijf een functie som_delers, die voor een gegeven positief geheel getal (dat als parameter aan de functie meegegeven wordt) de som van de echte delers van het getal teruggeeft. Zo moet de functie voor de waarde 9 het resultaat 4 (= $1 + 3$) teruggeven, en resulteert de waarde 10 in de som 8 (= $1 + 2 + 5$).

   ```
   def som_delers(n)
   ```

2. Gebruik de functie som_delers om een functie getalsoort te schrijven, die voor een gegeven positief geheel getal een string teruggeeft die aangeeft om welk soort getal het gaat: perfect, gebrekkig of overvloedig. Zo moet de functie voor de waarde 28 de tekenreeks perfect als resultaat teruggeven.

   ```
   def getalsoort(n)
   ```

Voorbeeld

```
>>> som_delers(28)
28
>>> getalsoort(28)
'perfect'
>>> som_delers(29)
1
>>> getalsoort(29)
'gebrekkig'
>>> som_delers(30)
42
>>> getalsoort(30)
'overvloedig'
```