Algorithm definition

New! An algorithm is a finite sequence of precise instructions for solving a problem.

Algorithms can be expressed in English or in more formalized descriptions like pseudocode or fully executable programs.

Sometimes, we can define algorithms whose output matches the rule for a function we already care about. Consider the (integer) logarithm function

$$logb : \{b \in \mathbb{Z} \mid b > 1\} \times \mathbb{Z}^+ \quad \to \quad \mathbb{N}$$

defined by

$$logb(\ (b, n)\ ) = \text{greatest integer } y \text{ so that } b^y \text{ is less than or equal to } n$$

Calculating integer part of base b logarithm

```
procedure logb(b,n: positive integers with b > 1)
i := 0
while n > b - 1
  i := i + 1
  n := n div b
return i {i holds the integer part of the base b logarithm of n}
```

Trace this algorithm with inputs $b = 3$ and $n = 17$

	b	n	i	$n > b - 1$?
Initial value	3	17		
After 1 iteration				
After 2 iterations				
After 3 iterations				

Compare: does the output match the rule for the (integer) logarithm function?

Base expansion algorithms

Two algorithms for constructing base b expansion from decimal representation

Most significant first: Start with left-most coefficient of expansion (highest value)

Informally: Build up to the value we need to represent in "greedy" approach, using units determined by base.

Calculating base b expansion, from left

```
procedure baseb1(n, b: positive integers with b > 1)
v := n
k := 1+ output of logb algorithm with inputs b and n
for i := 1 to k
   a_{k-i} := 0
   while v ≥ b^{k-i}
      a_{k-i} := a_{k-i} + 1
      v := v - b^{k-i}
return (a_{k-1}, ..., a_0){(a_{k-1} ... a_0)_b is the base b expansion of n}
```

Least significant first: Start with right-most coefficient of expansion (lowest value)

Idea: (when $k > 1$)

$$
\begin{aligned}
n &= a_{k-1}b^{k-1} + \cdots + a_1 b + a_0 \\
&= b(a_{k-1}b^{k-2} + \cdots + a_1) + a_0
\end{aligned}
$$

so $a_0 = n \textbf{ mod } b$ and $a_{k-1}b^{k-2} + \cdots + a_1 = n \textbf{ div } b$.

Calculating base b expansion, from right

```
procedure baseb2(n,b: positive integers with b > 1)
q := n
k := 0
while q ≠ 0
   a_k := q mod b
   q := q div b
   k := k + 1
return (a_{k-1}, ..., a_0){(a_{k-1} ... a_0)_b is the base b expansion of n}
```

Base conversion algorithm

Practice: write an algorithm for converting from base b_1 expansion to base b_2 expansion: