One of the oldest algorithms is probably the famous Euclidean algorithm. Its purpose is to compute the greatest common divisor (gcd) of two natural numbers.
That is, given $a, b \in \mathbb{N}$ the gcd is the largest natural number that divides a as well as b.
The existence of the gcd follows from the fact that a and b at least have 1 as common divisor.
This makes gcd being a function

$$gcd : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$

Let us assume $a > b$ and

$$gcd(a, b) = g \tag{1}$$

Then by Euclid's division lemma there exist unique numbers m_0, $r_0 < b$ such that

$$a = m_0 \cdot b + r_0 \tag{2}$$

If $r_0 = 0$ then trivially $g = b$. Otherwise, since a is divisible by g and b as well, r_0 can be written in the form

$$r_0 = n_0 \cdot g$$

In other words, g is a divisor of r_0. Moreover, it is the greatest common divisor of b and r_0:

$$gcd(b, r_0) = g \tag{3}$$

For this to see assume to the contrary $gcd(b, r_0) > g$. Then from the representation of a by (2) we would infer that $gcd(b, r_0)$ as well is a divisor of a and b. But this contradicts g being the greatest among all common divisors of a and b.
Comparing equations (1) and (3), we see they are for the same g but the latter involving numbers strictly lower than those of the first ($a > b$ and $b > r_0$). So, we have reduced the initial problem of finding the gcd for a and b to one that involves lower numbers. Exactly this can be exploited to formulate a recursive algorithm. The terminal condition is given by $r_0 = 0$ that produces a gcd like explained above.