Definition: Euclidean Algorithm

Given two positive integers m and n find their greatest common divisor, that is, the largest positive integer that evenly divides both m and n.

Divide m by n and let r be the remainder where $0 \leq r < n$ If $r = 0$, the algorithm terminates; n is the answer. Set $m \leftarrow n$, $n \leftarrow r$ and return to step 1

Algorithm** $Euclid_A lg$**isinputs** : n, mr← m % n **if** r $= 0$ **then** return r **end** return $Euclid_A lg$(n, r)**end Algorithm**$Euclid_A lg$**is

Correctness

Proof

- *Note that by the GCD invariant we have: $\gcd(m, n) = \gcd(n, r)$, then each time we go to step 3 this chain of equalities would expand by one see why we would be applying the quotient remainder theorem on n in the next iteration to obtain $n = qn_1 + r_1$, then we would have*

$$\gcd(n, m) = \gcd(n, r) = \gcd(n_1, r_1)$$

- *After finitely many iterations our algorithm get to the second step (read the termination proof) and say it's called with n_t, r_t (t for termination)*
- *It's in the second step so $r_t = 0$ and $n_t = \gcd(n_t, 0) = \gcd(n_t, r_t) = ... \gcd(n_1, r_1) = \gcd(n, r) = gcd(m, n)$ (the chain of equalities)*
- *Our output would be $n_t = \gcd(n, m)$, as required.*

■

Termination

The program terminates if $r = 0$, the value of n decreases by at least 1 after each iteration specified by the strict inequality from the quotient remainder theorem, therefore if n_k is the value of n after k iterations then $n_0, n_1, ...$ is a decreasing sequence of positive integers, and so it must be finite, therefore there is a $r \in \mathbb{N}$ such that the algorithm terminates on iteration r (as $n_r = 0$)