- How algorithmists write proofs?
- Time complexity analysis for Euclid.
- Recurrence equations
- Multiple return values in C.

Loop invariant: A statement that is true at the beginning of each iteration of the loop.

For Euclid's GCD:

The gcd of variables n and m is the same as the gcd of the input values.

- Initialization : Invariant is true initially
- Preservation : Invariant is true across iterations
- Termination : Use correctness of invariant to prove correctness of algorithm

Initialization: The set $\{n, m\}$ is the same as set of i/p values. So Invariant is true initially.

Preservation:

Use GCD theorem

let N, M be values of n and m at the beginning then n and m are (M, N%M) at the beginning of next iteration.

Termination

loop terminates when $m=0$ and in this case gcd is the value in n.

Why is Euclid's algo fast?

We define the **weight** of input (n, m) as $n+m$.

$(n, m) \to (m, r)$ s.t $n = qm + r$

$n \geq m$

Goal. Compare $(m+r)$ to $(n+m)$

$n+m \geq 2m$ $\quad$ $m+r < 2m$

We do a Proof by exhaustive cases.

1) $q \geq 2$ $\quad$ $n+m \geq 3m$

2) $q = 1$ $\quad$ Again split into two cases.

2a) $0 \leq r < m/2$: $m+r < \frac{3}{2}m$

2b) $r \geq \frac{m}{2}$

$n+m \to m+r$

1) $3m \to 2m$

2a) $2m \to \frac{3}{2}m$

2b) $\frac{5}{2}m \to 2m$

$n+m \geq m + \frac{m}{2} + m$

$= \frac{5}{2}m$

vs

$m+r < 2m$

It is true that.

$m+r \leq \frac{4}{5}(n+m)$

i/p weight

$$w \rightarrow \frac{4}{5}w \rightarrow \left(\frac{4}{5}\right)^2 w \rightarrow \cdots$$

Assume $w = \left(\frac{5}{4}\right)^k$. Then, we'll terminate in k steps. $k = \log_{5/4} w$

k is s.t.

$$\left(\frac{5}{4}\right)^k \leq w < \left(\frac{5}{4}\right)^{k+1}$$

So algo. terminates in at most $k+1 = \log_{5/4}(w)$ steps.

$$= \log_{5/4}(n+m)$$

$$t(w) \le t\left(\left\lceil \frac{4}{5} w \right\rceil\right) + 1$$

$$t(w) \le \underbrace{100}_{} \quad \text{for } w \le 5$$