# Termination

Recursion

$$x - \mathsf{0} \to x$$

$$\mathsf{s}(x) - \mathsf{s}(y) \to x - y$$

$$\mathsf{quot}(\mathsf{0}, \mathsf{s}(y)) \to \mathsf{0}$$

$$\mathsf{quot}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{s}(\mathsf{quot}(x - y, \mathsf{s}(y)))$$

$$\mathsf{0} + y \to y$$

$$\mathsf{s}(x) + y \to \mathsf{s}(x + y)$$

$$(x - y) - z \to x - (y + z)$$

$$
\begin{aligned}
\mathsf{le}(0, y) &\to \mathsf{true} \\
\mathsf{le}(\mathsf{s}(x), 0) &\to \mathsf{false} \\
\mathsf{le}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{le}(x, y) \\
\mathsf{minus}(0, y) &\to 0 \\
\mathsf{minus}(\mathsf{s}(x), y) &\to \mathsf{if}_{\mathsf{minus}}(\mathsf{le}(\mathsf{s}(x), y), \mathsf{s}(x), y) \\
\mathsf{if}_{\mathsf{minus}}(\mathsf{true}, \mathsf{s}(x), y) &\to 0 \\
\mathsf{if}_{\mathsf{minus}}(\mathsf{false}, \mathsf{s}(x), y) &\to \mathsf{s}(\mathsf{minus}(x, y)) \\
\mathsf{quot}(0, \mathsf{s}(y)) &\to 0 \\
\mathsf{quot}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{s}(\mathsf{quot}(\mathsf{minus}(x, y), \mathsf{s}(y)))
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{le}(\mathsf{s}(x), \mathsf{0}) &\to \mathsf{false} \\
\mathsf{le}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{le}(x, y) \\
\mathsf{pred}(\mathsf{s}(x)) &\to x \\
\mathsf{minus}(x, \mathsf{0}) &\to x \\
\mathsf{minus}(x, \mathsf{s}(y)) &\to \mathsf{pred}(\mathsf{minus}(x, y)) \\
\mathsf{gcd}(\mathsf{0}, y) &\to y \\
\mathsf{gcd}(\mathsf{s}(x), \mathsf{0}) &\to \mathsf{s}(x) \\
\mathsf{gcd}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{if}_{\mathsf{gcd}}(\mathsf{le}(y, x), \mathsf{s}(x), \mathsf{s}(y)) \\
\mathsf{if}_{\mathsf{gcd}}(\mathsf{true}, \mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{gcd}(\mathsf{minus}(x, y), \mathsf{s}(y)) \\
\mathsf{if}_{\mathsf{gcd}}(\mathsf{false}, \mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{gcd}(\mathsf{minus}(y, x), \mathsf{s}(x))
\end{aligned}
$$

$$
\begin{aligned}
\mathsf{le}(\mathsf{s}(x),\mathsf{s}(y)) &\to \mathsf{le}(x,y)\\
\mathsf{app}(\mathsf{nil},y) &\to y\\
\mathsf{app}(\mathsf{add}(n,x),y) &\to \mathsf{add}(n,\mathsf{app}(x,y))\\
\mathsf{low}(n,\mathsf{nil}) &\to \mathsf{nil}\\
\mathsf{low}(n,\mathsf{add}(m,x)) &\to \mathsf{if}_{\mathsf{low}}(\mathsf{le}(m,n),n,\mathsf{add}(m,x))\\
\mathsf{if}_{\mathsf{low}}(\mathsf{true},n,\mathsf{add}(m,x)) &\to \mathsf{add}(m,\mathsf{low}(n,x))\\
\mathsf{if}_{\mathsf{low}}(\mathsf{false},n,\mathsf{add}(m,x)) &\to \mathsf{low}(n,x)\\
\mathsf{high}(n,\mathsf{nil}) &\to \mathsf{nil}\\
\mathsf{high}(n,\mathsf{add}(m,x)) &\to \mathsf{if}_{\mathsf{high}}(\mathsf{le}(m,n),n,\mathsf{add}(m,x))\\
\mathsf{if}_{\mathsf{high}}(\mathsf{true},n,\mathsf{add}(m,x)) &\to \mathsf{high}(n,x)\\
\mathsf{if}_{\mathsf{high}}(\mathsf{false},n,\mathsf{add}(m,x)) &\to \mathsf{add}(m,\mathsf{high}(n,x))\\
\mathsf{quicksort}(\mathsf{nil}) &\to \mathsf{nil}\\
\mathsf{quicksort}(\mathsf{add}(n,x)) &\to \mathsf{app}(\mathsf{quicksort}(\mathsf{low}(n,x)),\\
&\qquad\qquad \mathsf{add}(n,\mathsf{quicksort}(\mathsf{high}(n,x))))
\end{aligned}
$$

# Apply

```
apply(t,σ) :=
    if var?(t)
    then if σ={}
        then t
        else let {x↦u}∪σ'=σ in
            if t=x
            then u
            else apply(t,σ')
    else let f(t1,...,tn)=t in
        f(apply(t1,σ),...,apply(tn,σ))
```

# Occur?

```
occur?(x,t) :=
    if var?(t)
    then (x=t)
    else let f(t1,...,tn)=t in
        occur?(x,t1) ∨...∨ occur?(x,tn)
```

# Unify

```
unify(s,t) :=
  if var?(s)
  then if var?(t)
        then if s=t then {} else {s↦t}
        else if occur?(s,t)
              then fail
              else {s↦t}
  else let f(s1,...,sm) = s & g(t1,...,tn) = t in
        if f≠g
        then fail
        else if m=0 [assuming m=n]
              then {}
              else let σ = unify(s1,t1) in
                    let τ = unify(apply(f(s2,...,sm), σ),apply(f(t2,...,tn), σ)) in
                    τ ∪ στ [composition of substitutions....]
```

# Primitive Recursion

- f(n,x,...,z) :=

  if n=0

  then g(x,...,z)

  else h(f(n-1,x,...,z),n-1,x,...,z)

# Inductive Definitions

- Constructors
  - 0, s(0), s(s(0)), ...
  - e, a(e), b(e), a(a(e)), ...
  - e, b(e,e), b(b(e,e),e), ...

# Structural Induction

- a(x,*y*) := if x=() then *y* else c(hd(x),a(tl(x),*y*))
- r(x) := if x=() then () else a(r(tl(x)),c(hd(x),()))

# Functions

- Basic (e.g. arithmetic, boolean)
- Constructors (e.g. lists, trees)
- Conditional (if c then a else b)
- Defined (recursively, perhaps)

# Definitions

- $f(x,y,\ldots,z) :\approx t[x,y,\ldots,z]$
- $e(m,n) :\approx \text{if } n=0 \text{ then } 1 \text{ else } m\times e(m,n-1)$

# Evaluations

- $\mathrm{if}(T,x,y) \Rightarrow x$
- $\mathrm{if}(F,x,y) \Rightarrow y$
- $\mathrm{if}(c,x,y) \Rightarrow \mathrm{if}(c',x,y)$
- $f(x,y) \Rightarrow t[x,y]$
- $f(x,y) \Rightarrow f(x',y)$
- $f(x,y) \Rightarrow f(x,y')$

# Inner/Outer

- if(T,x,y) ⇨ x
- if(F,x,y) ⇨ y
- if(c,x,y) ⇨ if(c',x,y)
- f(x,y) ⇨ t[x,y]
- f(x,y) ⇨ f(x',y)
- f(x,y) ⇨ f(x,y')

# Inner & Outer

- N: normative; nothing above
- A: applicative; nothing below
- I: inner; something above (not normal)
- O: outer; something below

# 91 Example

- f(x) := if x>100

  then x-10

  else f(f(x+11))

# Example

- f(x,y) := if x=0

  then 2

  else f(x-1,f(x+y,y))

# Example

- f(x,y) := if x=0

  then 0

  else if x=1

  then f(0,f(1,y))

  else f(x-2,y+1)

# Example

- $f(1,1) = f(0,f(1,1)) = ???$

# In vs. Out

- If any computation is terminating, then outermost (normal order) is terminating.
- If any computation is non-terminating, then innermost (applicative order) is non-terminating.

# Normal is Very Good

- Suppose not
- Consider minimal counterexample
- u NNNNINNIINNNNIII v ; v value
- I N ≈ I O ⊆ N A*
- So: u N…N I…I v
- But can't have Iv, so u N* v

# Applicative is Very Bad

- If u O v, then
  - there are u' v' v'' such that
  - $u \ A^! \ u' \ A \ v' \ A^! \ v''$
  - $v \ A^* \ v' \ A^! \ v''$
  - $A^!$ means as much as possible