1-1

```
function zero ( F, Z ) { return Z; }
function add1 ( N ) { return function ( F, Z ) { return N(F, F(Z)); }
two =
add1 ( add1 ( zero )) ;

two ( function ( X ) { console.print( "Hey!" ); } , 0 );

two ( function ( X ) { return X + 1; } , 0 );
```

Normative : • Theories are given
• Construct models

Descriptive : • Models are given
• Find theories that describe them

ISWIM $M, N, L, K = X \mid (\lambda X. M) \mid (M\ N) \mid b \mid (o^n\ M \ldots)$

b = constants

o^n = are n-ary primitives

$FV(b) = \{\}$

$FV(o^n\ M_1 \ldots M_n) = FV(M_1) \cup \cdots \cup FV(M_n)$

$b = \{\text{true}, \text{false}\} \cup N$ (= natural number)

$o^1 = \{\text{add1}, \text{sub1}, \text{iszero}\}$

$o^2 = \{+, -, *, \div, \%, \uparrow, <, >, =, \ldots\}$

$V, U, W = b \mid (\lambda X. M)$ (values or answers)

$\beta_v : (\lambda X. M)\ V \rightarrow M[X \leftarrow V]$ (in ISWIM)

$\beta : (\lambda X. M)\ N \rightarrow M[X \leftarrow N]$ (in λ)

$(\lambda X. X + X)\ (5 + 5)$ $\quad \beta \quad (5+5) + (5+5)$

$(5+5) \rightarrow 10$ $\quad \beta_v \quad 10 + 10$

4-2

δ : $(o^n\ V_1 \ldots V_n) \rightarrow \delta^n(o^n, V_1, \ldots, V_n)$

where δ^n is a parameter (like b and o^n) of language

$\delta^1(\text{add1}, 5) = 6$ $\qquad$ $\delta^1(\text{not}, \text{true}) = \text{false}$

$\delta^2(+, 5, 10) = 15$ $\qquad$ etc.

$v = \beta_v \cup \delta$ $\qquad$ $\twoheadrightarrow_v$ = refl-trans clo of $\rightarrow_v$

$\rightarrow_v$ = compatible closure of v $\qquad$ $=_v$ = sym clo. of $\twoheadrightarrow_v$

$\text{eval}_v(M) = b$ $\qquad$ if $M =_v b$

$\qquad$ 'function $\qquad$ if $M =_v \lambda X. N$

(i.e. we observe which function is returned.)

eval_v is partial (e.g. $\text{eval}_v(\Omega) = \bot$)

we call non-existents, divergence (Ω diverges)

$(\lambda X. (X\ 5))\ 0 \rightarrow 0\ 5 \not\rightarrow$ — "stuck"

$\Omega \rightarrow \Omega$ — divergence

both are partial results of eval_v

$Y_v = \lambda F.$ $\qquad$ call-by-value Y-combinator

$\quad \lambda X.$

$\quad (((\lambda G. F\ (\lambda Y. (G\ G)\ Y)))$

$\quad\ (\lambda G\ F\ (\lambda Y. (G\ G)\ Y)))\ X)$

Program $\xrightarrow{\text{semantics}}$ meaning

||

implementation $\qquad$ answer