Programming constructs in λ calc.

$\lambda x.\ M$

$((\lambda x. \lambda y.\ M)\ a)\ b \qquad f(a,b)$

"Currying"

$(\lambda x.\ \lambda y.\ x + y)\ 3$ function specialization

$\longrightarrow (\lambda y.\ 3 + y)\ 2$

$\longrightarrow 3 + 2$

Church Boolean

$$\left.\begin{aligned} tru &= \lambda \underline{t}.\ \lambda \underline{f}.\ t \\ fls &= \lambda t.\ \lambda f.\ f \end{aligned}\right\}$$

$$NOT = \lambda x. (x\ \underbrace{fls}_{①})\ \underbrace{tru}_{②}$$

tru returns ①

fls returns ②

conditional

if (b) then v else w

$$\text{cond} = \lambda l.\ \lambda m.\ \lambda n.\ \underbrace{l\ m\ n}$$

$\underline{\text{cond}}\ \underline{\text{tru}}\ v\ w$

$\rightarrow (\lambda m.\ \lambda n.\ \text{tru}\ m\ n)\ v\ w$

fls v w

$\rightarrow\ \rightarrow$ tru v w

$\rightarrow v$

$\rightarrow w$

$$\text{AND} = \lambda p.\ \lambda q.\ (p\ q)\ \text{fls}$$

$(p\ q)\ \overset{!!!}{p}$

$$\text{OR} = \lambda p.\ \lambda q.\ (p\ \text{tru})\ q$$

$(p\ p)\ q$

$$\text{pair} = \lambda f.\ \lambda s.\ \lambda b.\ b\ f\ s$$

getter

$$\text{fst} = \lambda p.\ p\ \text{tru}$$

$$\text{snd} = \lambda p.\ p\ \text{fls}$$

e.g. (v, w)

$\text{pair}\ v\ w \longrightarrow \underbrace{\lambda b.\ b\ v\ w}$

$\text{fst}\ (\text{pair}\ v\ w) \longrightarrow (\lambda p.\ p\ \text{tru})\ \underbrace{(\lambda b.\ v\ w)}$

$\longrightarrow (\lambda b.\ b\ v\ w)\ \text{tru}$

$\longrightarrow$ tru v w

$\longrightarrow$ v

Church numerals

$c_0 = \lambda s. \lambda z.\ z$ — zero

$c_1 = \lambda s. \lambda z.\ s\ z$ — successor

$c_2 = \lambda s. \lambda z.\ s(s\ z)$

$c_3 = \lambda s. \lambda z.\ s(s(s\ z))$

0000
0001
0010

$fls = \lambda x. \lambda y.\ y$

$\lambda s. \lambda z.\ z$

$\cancel{\lambda s. \lambda z.}\ s(s\ z)$

s

$inc = \lambda n.\ \underbrace{\lambda s. \lambda z.}_{\text{rebuild } \lambda s.\lambda z}\ s(\underbrace{n\ s\ z}_{\text{demolish } \lambda s.\lambda z})$

adds an additional s in n

$inc\ \underbrace{(\lambda s'. \lambda z'.\ z')}_{c_0}$

$\longrightarrow \lambda s. \lambda z.\ s\ (\underbrace{(\cancel{\lambda s'. \lambda z'.}\ z')}_{c_0}\ s\ z)$

$\longrightarrow \lambda s. \lambda z.\ s\ z$

$$\text{plus} = \lambda m.\lambda n.\lambda s.\lambda z.\ (m\ s)\ (n\ s\ z)$$

$m = 2$

$= \lambda s.\lambda z.\ s\ (s\ z)$

$n = 1$

$= \lambda s.\lambda z.\ s\ z$

$m\ s$

$\rightarrow \lambda z.\ s\ (s\ z)$

$n\ s\ z$

$\rightarrow s\ z$

$$s\ (s\ (s\ z))$$

$$\text{times} = \lambda m.\lambda n.\ m\ (\text{plus}\ n)\ c_0$$

$$s\ (s\ (s \ldots\ \ldots\ z))$$

$$\text{plus2} = \lambda m.\lambda n.\ m\ \text{inc}\ n$$

$$\text{inc}\ (\text{inc}\ (\text{inc} \ldots\ n)))$$

$$s\ \ldots\ldots\ s\ z$$

n

$\text{inc} \longleftrightarrow \text{inc}$

m times

$$\text{isZero} = \lambda n.\ n\ \underbrace{(\text{AND fls})}_{\lambda x.\,\text{fls}}\ \text{tru}.$$

$$c_0 = \lambda s.\,\lambda z.\,z$$

$$c_0\ \underbrace{\lambda x.x}\ \text{tru}$$

$$c_1 = \lambda s.\,\lambda z.\ s\ z$$

$$c_1\ \underbrace{\lambda x.x}\ \text{tru}$$

$$(\text{AND fls})\ \text{tru}$$

$$\to \text{fls}$$

$$\underbrace{(\text{AND fls})\ (\underbrace{(\text{AND fls})\ \text{tru}}_{\text{fls}})}_{\text{fls}}$$

Exponentiation

$$m^n$$

$$\exp = \lambda m.\,\lambda n.\ n\ m$$

OCaml

de Brujin indices