1 Part 1

$$
\begin{array}{rcl}
n & ::= & 0 \mid add1(n) \\
b & ::= & \text{true} \mid \text{false} \\
v & ::= & n \mid b \\
e & ::= & v \mid \lambda x.e \mid e_1\ e_2 \mid x \mid \text{if } e \text{ then } e_1 \text{ else } e_2 \\
 & & \mid \{x \mapsto v, e\} \mid \{\} \mid e.x \\
 & & \mid nfold\langle e_1, e_2, e_3\rangle
\end{array}
$$

The grammar has 4 parts: nats, modeled as 0 or add1 to an existing natural number; bool, which can be true or false; values, which are nats or bools; and expressions, which can be values, lambdas, applications, variable lookups, if statements, dictionaries, projections, or nat-folds. Values and empty dictionaries are expression introduction forms, whereas the rest are elimination forms.

The BNF for nats is well formed because 0 is a base case with no self-references and the n in $add1(n)$ is smaller than the form $add1(n)$. Booleans are well formed because there are two base cases with no self-references, and no other forms. Expressions have values and empty dictionaries as base cases, and in every judgement that produces an expression, some extra symbols are required to be present as well as sub-expression(s), meaning that the sub-expression(s) must contain less symbols, and therefore be smaller.

2 Part 2

$\boxed{e \to e}$

$$
\begin{array}{rcl}
\{x \mapsto v, e\}.x & \to & v \\
\{x \mapsto v, e\}.y & \to & e.y \text{ where } x \neq y \\
\text{if true then } e_1 \text{ else } e_2 & \to & e_1 \\
\text{if false then } e_1 \text{ else } e_2 & \to & e_2 \\
0 + n & \to & n \\
add1(n) + n' & \to & n + add1(n') \\
(\lambda x.e_1)\ e_2 & \to & e_1[e_2/x] \\
nfold\langle 0, e_2, e_3\rangle & \to & e_2 \\
nfold\langle add1(n), e_2, e_3\rangle & \to & (e_3\ n)\ nfold\langle n, e_2, e_3\rangle \\
nfold\langle e_1, e_2, e_3\rangle & \to & \text{false where } e_1 \notin n
\end{array}
$$

$\boxed{e \longrightarrow^* e}$

$$\text{(refl)}\ \frac{}{e \longrightarrow^* e} \qquad \text{(trans)}\ \frac{e_1 \longrightarrow e_2 \quad e_2 \longrightarrow^* e_3}{e_1 \longrightarrow^* e_3}$$

$$\text{(cong-if)}\ \frac{e \longrightarrow^* e'}{\text{if } e \text{ then } e_1 \text{ else } e_2 \longrightarrow^* \text{if } e' \text{ then } e_1 \text{ else } e_2} \qquad \text{(cong-add)}\ \frac{e_1 \longrightarrow^* e_1' \quad e_2 \longrightarrow^* e_2'}{e_1 + e_2 \longrightarrow^* e_1' + e_2'}$$

$$\text{(cong-proj)}\ \frac{e \longrightarrow^* e'}{e.x \longrightarrow^* e'.x} \qquad \text{(cong-dict)}\ \frac{e \longrightarrow^* e'}{\{x \mapsto v, e\} \longrightarrow^* \{(x \mapsto v), e'\}} \qquad \text{(cong-app)}\ \frac{e_1 \longrightarrow^* e_1'}{e_1\ e_2 \longrightarrow^* e_1'\ e_2}$$

$$\text{(cong-nat-fold)}\ \frac{e_1 \longrightarrow^* e_1' \quad e_2 \longrightarrow^* e_2' \quad e_3 \longrightarrow^* e_3'}{nfold\langle e_1, e_2, e_3\rangle \longrightarrow^* nfold\langle e_1', e_2', e_3'\rangle}$$

$\boxed{eval(e) = v}$

$$\frac{e \longrightarrow^* v}{eval(e) = v}$$

3 Part 3

Not all syntactic expressions are well-defined. For example, $true + false$ can not be evaluated to a value since no reduction step can be taken on $true + false$, and $true + false$ is not a value.

4 Part 4

Dictionary: $\{(a \mapsto 5), \{(b \mapsto 120), \{(c \mapsto \text{true}), \{\}\}\}\}$

$$eval(\{(a \mapsto 5), \{(b \mapsto 120), \{(c \mapsto \text{true}), \{\}\}\}\}.b) = 120$$

Proof:

$$\dfrac{\dfrac{}{\{a \mapsto 5, \{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}\}.b \longrightarrow \{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}.b} \quad \dfrac{\dfrac{\overline{\{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}.b \longrightarrow 120} \quad \overline{120 \longrightarrow^* 120}}{\{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}.b \longrightarrow^* 120}}{\{a \mapsto 5, \{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}\}.b \longrightarrow^* 120}}{eval(\{a \mapsto 5, \{b \mapsto 120, \{c \mapsto \text{true}, \{\}\}\}\}.b = 120}$$

5 Part 5

$is\text{-}zero? = \lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle$

$eval(is\text{-}zero?\ 0) = \text{true}$:

$$D_1 = \overline{(\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 0 \longrightarrow nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle)}$$

$$D_2 = \dfrac{nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle) \longrightarrow \text{true} \quad \text{true} \longrightarrow^* \text{true}}{nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle) \longrightarrow^* 0}$$

$$\dfrac{\dfrac{D_1 \quad D_2}{(\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 0 \longrightarrow^* 0}}{eval(is\text{-}zero?\ 0) = \text{true}}$$

$eval(is\text{-}zero?\ 1) = \text{false}$:

$$\begin{aligned}
eval(is\text{-}zero?\ 1) &= eval((\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 1) && (0)\\
(\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 1 &\longrightarrow nfold\langle 1, \text{true}, \lambda y.\lambda z.\text{false}\rangle && (1)\\
nfold\langle 1, \text{true}, \lambda y.\lambda z.\text{false}\rangle &\longrightarrow ((\lambda y.\lambda z.\text{false})\ 0)\ nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle && (2)\\
((\lambda y.\lambda z.\text{false})\ 0)\ nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle &\longrightarrow (\lambda z.\text{false})\ nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle && (3)\\
(\lambda z.\text{false})\ nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle &\longrightarrow \text{false} && (4)\\
&\text{(eliding symmetry)}\\
(\lambda z.\text{false})\ nfold\langle 0, \text{true}, \lambda y.\lambda z.\text{false}\rangle &\longrightarrow^* \text{false} && (5)\\
&\text{(eliding 3 steps of transitivity)}\\
(\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 1 &\longrightarrow^* \text{false} && (7)\\
eval((\lambda x.nfold\langle x, \text{true}, \lambda y.\lambda z.\text{false}\rangle)\ 1) &= \text{false} && (8)\\
eval(is\text{-}zero?\ 1) &= \text{false} && (9)
\end{aligned}$$

$eval(is\text{-}zero?\ \text{true}) = \text{false}$:

$$
\begin{aligned}
eval(is\text{-}zero?\ \mathsf{true}) &= eval((\lambda x.nfold\langle x, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle)\ \mathsf{true}) && (0)\\
(\lambda x.nfold\langle x, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle)\ \mathsf{true} &\longrightarrow nfold\langle \mathsf{true}, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle && (1)\\
nfold\langle \mathsf{true}, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle &\longrightarrow \mathsf{false} && (2)\\
\text{(eliding symmetry)} & && (3)\\
nfold\langle \mathsf{true}, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle &\longrightarrow^* \mathsf{false} && (4)\\
\text{(eliding transitivity)} & && \\
(\lambda x.nfold\langle x, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle)\ \mathsf{true} &\longrightarrow^* \mathsf{false} && (5)\\
eval((\lambda x.nfold\langle x, \mathsf{true}, \lambda y.\lambda z.\mathsf{false}\rangle)\ \mathsf{true}) &= \mathsf{false} && (6)\\
eval(is\text{-}zero?\ \mathsf{true}) &= \mathsf{false} && (7)
\end{aligned}
$$

6 Part 6

$+ = \lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle$

$eval(+\ 0\ 1) = 1$: (Derivation done in pieces to fit on page)

$$D_0 = \frac{\overline{nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle \longrightarrow 0} \quad \overline{0 \longrightarrow^* 0}}{nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle \longrightarrow^* 0}$$

$$D_1 = \frac{nfold\langle 0, nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle \longrightarrow nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle \quad D_0}{nfold\langle 0, nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle \longrightarrow^* 0}$$

$$D_2 = \frac{\lambda y.nfold\langle 0, nfold\langle u, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0 \longrightarrow nfold\langle 0, nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle) \quad D_1}{\lambda y.nfold\langle y, nfold\langle 0, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0 \longrightarrow^* 0}$$

$$D_3 = \overline{\lambda x.\lambda y.nfold\langle y, nfold\langle x, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle)\ 0\ 0 \longrightarrow \lambda y.nfold\langle 0, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0}$$

$$D_4 = \frac{D_2}{\lambda y.nfold\langle 0, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle)\ 0 \longrightarrow^* 0}$$

$$\frac{\frac{D_3 \quad D_4}{\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0\ 0 \longrightarrow^* 0}}{eval(+\ 0\ 0) = 0}$$

Lemma 1: $\forall n \in Nats. nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle \longrightarrow^* add1(n)$

Proof:

$$\begin{array}{rcl} nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow & (\lambda a.\lambda b.add1(b))\ 0\ n \\ (\lambda a.\lambda b.add1(b))\ 0\ 0 & \longrightarrow & \lambda b.add1(b))\ n \\ \lambda b.add1(b))\ 0 & \longrightarrow & add1(n) \\ \text{(eliding symmetry and transitivity)} & & \\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(n) \end{array}$$

Lemma 2: $\forall n \in Nats. nfold\langle 2, n, \lambda a.\lambda b.add1(b)\rangle \longrightarrow^* add1(add1(n))$

Proof:

$$
\begin{array}{rcl}
nfold\langle 2, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow & (\lambda a.\lambda b.add1(b))\ 1\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle \\
(\lambda a.\lambda b.add1(b))\ 1\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow & \lambda b.add1(b)\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle \\
\lambda b.add1(b)\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow & add1(nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle) \\
\text{(from lemma 1)} & & \\
\lambda b.add1(b)\ nfold\langle 1, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(add1(n)) \\
\text{(eliding symmetry and transitivity)} & & \\
nfold\langle 2, n, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(add1(n))
\end{array}
$$

$eval(+\ 0\ 1) = 1$:

$$
\begin{array}{rcll}
eval(+\ 0\ 1) & = & eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0\ 1) & (0) \\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0\ 1 & \longrightarrow & \lambda y.nfold\langle 0, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1 & (1) \\
\lambda y.nfold\langle 0, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1 & \longrightarrow & nfold\langle 0, nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & (2) \\
nfold\langle 0, nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow & nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle & (3) \\
\text{(using lemma 1)} & & & \\
nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(0) & (4) \\
\text{(eliding 2 steps of transitivity)} & & & \\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0\ 1 & \longrightarrow^* & 1 & \\
eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 0\ 1) & = & 1 & \\
eval(+\ 0\ 1) & = & 1 &
\end{array}
$$

$eval(+\ 1\ 1) = 2$:

$$
\begin{array}{rcll}
eval(+\ 1\ 1) & = & eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 1) & (0) \\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 1 & \longrightarrow & \lambda y.nfold\langle 1, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1 & (1) \\
\lambda y.nfold\langle 1, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1 & \longrightarrow & nfold\langle 1, nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & (2) \\
\text{(using lemma 1)} & & & \\
nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(0) & (3) \\
nfold\langle 1, nfold\langle 1, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & nfold\langle 1, 1, \lambda a.\lambda b.add1(b)\rangle & (4) \\
\text{(using lemma 1)} & & & \\
nfold\langle 1, 1, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(1) & (5) \\
\text{(eliding 2 steps of transitivity)} & & & \\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 1 & \longrightarrow^* & 2 & (6) \\
eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 1) & = & 2 & (7) \\
eval(+\ 1\ 1) & = & 2 &
\end{array}
$$

$eval(+\ 1\ 2) = 3$:

$$
\begin{array}{rcll}
eval(+\ 1\ 2) & = & eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 2) & (0)\\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 2 & \longrightarrow & \lambda y.nfold\langle 1, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2 & (1)\\
\lambda y.nfold\langle 1, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2 & \longrightarrow & nfold\langle 1, nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & (2)\\
\multicolumn{4}{c}{\text{(using lemma 2)}}\\
nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(add1(0)) & (3)\\
nfold\langle 1, nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & nfold\langle 1, 2, \lambda a.\lambda b.add1(b)\rangle & (4)\\
\multicolumn{4}{c}{\text{(using lemma 1)}}\\
nfold\langle 1, 2, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(2) & (5)\\
\multicolumn{4}{c}{\text{(eliding 2 steps of transitivity)}}\\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 2 & \longrightarrow^* & 3 & (6)\\
eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 1\ 2) & = & 3 & (7)\\
eval(+\ 1\ 2) & = & 3 &
\end{array}
$$

$eval(+\ 2\ 2) = 4$:

$$
\begin{array}{rcll}
eval(+\ 2\ 2) & = & eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2\ 2) & (0)\\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2\ 2 & \longrightarrow & \lambda y.nfold\langle 2, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2 & (1)\\
\lambda y.nfold\langle 2, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2 & \longrightarrow & nfold\langle 1, nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & (2)\\
\multicolumn{4}{c}{\text{(using lemma 2)}}\\
nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(add1(0)) & (3)\\
nfold\langle 2, nfold\langle 2, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & nfold\langle 1, 2, \lambda a.\lambda b.add1(b)\rangle & (4)\\
\multicolumn{4}{c}{\text{(using lemma 1)}}\\
nfold\langle 2, 2, \lambda a.\lambda b.add1(b)\rangle & \longrightarrow^* & add1(add1(2) & (5)\\
\multicolumn{4}{c}{\text{(eliding 2 steps of transitivity)}}\\
\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2\ 2 & \longrightarrow^* & 4 & (6)\\
eval(\lambda x.\lambda y.nfold\langle x, nfold\langle y, 0, \lambda a.\lambda b.add1(b)\rangle, \lambda a.\lambda b.add1(b)\rangle\ 2\ 2) & = & 4 & (7)\\
eval(+\ 2\ 2) & = & 4 &
\end{array}
$$