1 Parameter Reduction in Primitive Recursive Definitions
We aim to prove that the following recursion scheme:

\[
\begin{align*}
h(x, 0) &= f(x), \\
h(x, t + 1) &= g(h(x, t))
\end{align*}
\]

that makes use of two **unary functions** \( f \) and \( g \) (added to a method for reducing the number of parameters) suffices to generate the the set of primitive recursive functions.
The standard primitive recursion scheme involves defining $h$ starting from the functions $f$ and $g$ as follows. Let $h : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ be a function of $n$ parameters defined by primitive recursion, where $n > 1$.

\[
\begin{align*}
  h(x_1, \ldots, x_n, 0) &= f(x_1, \ldots, x_n), \\
  h(x_1, \ldots, x_n, t + 1) &= g(t, h(x_1, \ldots, x_n, t), x_1, \ldots, x_n)
\end{align*}
\]

starting with the functions $f : \mathbb{N}^n \rightarrow \mathbb{N}$ and $g : \mathbb{N}^{n+2} \rightarrow \mathbb{N}$. The goal is to obtain functions that depend on fewer parameters which allow us to reconstitute the original functions.
Define the functions $\tilde{h} : \mathbb{N}^n \rightarrow \mathbb{N}$, $\tilde{f} : \mathbb{N}^{n-1} \rightarrow \mathbb{N}$, and $\tilde{g} : \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ as:

$$\tilde{h}(x_1, \ldots, x_{n-1}, t) = h(x_1, \ldots, x_{n-2}, \ell(x_{n-1}), r(x_{n-1}), t)$$
$$\tilde{f}(x_1, \ldots, x_{n-2}, x_{n-1}) = f(x_1, \ldots, x_{n-2}, \ell(x_{n-1}), r(x_{n-1}))$$
$$\tilde{g}(t, u, x_1, \ldots, x_{n-1}) = g(t, h(x_1, \ldots, \ell(x_{n-1}), r(x_{n-1}), t), x_1, \ldots, \ell(x_{n-1}), r(x_{n-1}))$$

Note that $\tilde{h}$, $\tilde{f}$, and $\tilde{g}$ have one less argument compared to $h$, $f$, $g$, respectively.
Then, we can write

\[ \tilde{h}(x_1, \ldots, x_{n-1}, 0) = \tilde{f}(x_1, \ldots, x_{n-1}) \]
\[ \tilde{h}(x_1, \ldots, x_{n-1}, t + 1) = \tilde{g}(t, \tilde{h}(x_1, \ldots, x_{n-1}, t), x_1, \ldots, x_{n-1}). \]

CLAIM: The original function \( h \) can be retrieved from the equation:

\[ h(x_1, \ldots, x_n, t) = \tilde{h}(x_1, \ldots, x_{n-2}, \langle x_{n-1}, x_n \rangle, t). \]
Indeed, we have

\[ h(x_1, \ldots, x_{n-1}, x_n, 0) = \tilde{h}(x_1, \ldots, x_{n-2}, \langle x_{n-1}, x_n \rangle, 0) \]

and

\[
\begin{align*}
    h(x_1, \ldots, x_{n-1}, x_n, t + 1) & = \tilde{h}(x_1, \ldots, x_{n-2}, \langle x_{n-1}, x_n \rangle, t) \\
    & = h(x_1, \ldots, x_{n-2}, x_{n-1}, x_n, t).
\end{align*}
\]
By iterating this process we can reduce the number of parameters to 1, that is, to recursions of the form:

\[
\begin{align*}
    h(x, 0) &= f(x), \\
    h(x, t + 1) &= g(t, h(x, t), x).
\end{align*}
\]
A further simplification can be achieved using the pairing function. Define

$$\tilde{h}(x, t) = \langle h(x, t), \langle x, t \rangle \rangle.$$

Then, we have:

$$\tilde{h}(x, 0) = \langle h(x, 0), \langle x, 0 \rangle \rangle = \langle f(x), \langle x, 0 \rangle \rangle = \tilde{f}(x),$$

$$\tilde{h}(x, t + 1) = \langle h(x, t + 1), \langle x, t + 1 \rangle \rangle$$
$$= \langle g(t, h(x, t), x), \langle x, t + 1 \rangle \rangle$$
$$= \tilde{g}(\tilde{h}(x, t)),$$

where

$$\tilde{g}(u) = \langle g(r(r(u)), \ell(u), \ell(r(u)), \langle \ell(r(u)), r(r(u)) + 1 \rangle) \rangle.$$

The original function can be retrieved from $\tilde{h}$ as $h(x, t) = \ell(\tilde{h}(x, t))$. 
Thus, we reached the recursion scheme:

\[
\begin{align*}
    h(x, 0) &= f(x), \\
    h(x, t + 1) &= g(h(x, t))
\end{align*}
\]
Recursion with no parameters of the form

\[
\begin{align*}
  h(0) &= k, \\
  h(t + 1) &= g(t, h(t))
\end{align*}
\]

can also be readily put in this form. The constant function \(k\) can be obtained by \(k\) compositions of \(s(x)\) beginning with \(n(x)\). Furthermore, define \(\hat{g}\) as \(\hat{g}(h(t)) = g(t, h(t))\). This will yield

\[
\begin{align*}
  h(0) &= k, \\
  h(t + 1) &= \hat{g}(h(t)).
\end{align*}
\]