Let $n$ be a natural constant, and let $x$ be a natural variable. Then
\[ x' = n^3 \iff x := n. \quad x' = x \times n. \quad x' = x \times n \]

Proof:
\[
x := n. \quad x' = x \times n. \quad x' = x \times n \quad \text{definition of sequential composition}
\]
\[
\equiv x := n. \quad \exists x'. \quad x' = x \times n \land x' = x' \times n \quad \text{one-point}
\]
\[
\equiv x := n. \quad x' = x \times n \times n \quad \text{substitution law}
\]
\[
\equiv x' = n \times x \times n \quad \text{arithmetic}
\]
\[
\equiv x' = n^3
\]

Now we have only one specification to refine, namely $x' = x \times n$, and it's a multiplication, which is easier than cubing. We'll have to use repeated addition, so we have to start $x$ at $0$, and then keep adding $n$. How many times do we add $n$? We add it $x$ times, but that's $x$ before we initialized it to $0$. So we have to save the value of $x$ before we initialize it to $0$, and we introduce natural variable $y$ for that.
\[
x' = x + y \times n \quad \text{says the final value } x' \text{ is the sum so far, that's } x, \text{ plus } y \text{ more values of } n.
\]

Proof:
\[
y := x. \quad x := 0. \quad x' = x + y \times n \quad \text{substitution law}
\]
\[
\equiv y := x. \quad x' = 0 + y \times n \quad \text{simplify, then substitution law}
\]
\[
\equiv x' = x \times n
\]

Now the last refinement is straightforward.
\[
x' = x + y \times n \iff \text{if } y = 0 \text{ then ok else } x := x + n. \quad y := y - 1. \quad x' = x + y \times n \quad \text{fi}
\]

Proof:
\[
\text{if } y = 0 \text{ then ok else } x := x + n. \quad y := y - 1. \quad x' = x + y \times n \quad \text{fi} \quad \text{expand ok, substitution twice}
\]
\[
\equiv \text{if } y = 0 \text{ then } x' = x \land y = y \text{ else } x' = x + n + (y - 1) \times n \quad \text{context, simplify}
\]
\[
\equiv \text{if } y = 0 \text{ then } x' = x + y \times n \land y = y \text{ else } x' = x + y \times n \quad \text{monotonicity}
\]
\[
\equiv \text{if } y = 0 \text{ then } x' = x + y \times n \text{ else } x' = x + y \times n \quad \text{case idempotent}
\]
\[
\equiv x' = x + y \times n
\]

Adding recursive time, we need to put $t := t + 1$ just before the recursive call. Since $t$ goes up 1 just when $y$ goes down 1, we see that the time must be $y$. So that last refinement becomes
\[
x' = x + y \times n \land t' = t + y
\]

We recalculate the refinement of $x' = x \times n$ with timing, and we find
\[
y := x. \quad x := 0. \quad x' = x + y \times n \land t' = t + y
\]
\[
\equiv x' = x \times n \land t' = t + x
\]

We recalculate the refinement of $x' = n^3$ with timing, and we find
\[
x := n. \quad x' = x \times n \land t' = t + x. \quad x' = x \times n \land t' = t + x
\]
\[
\equiv x' = n^3 \land t' = t + n^2 + n
\]

We have calculated the timing for the solution to be $n^2 + n$, which wasn't obvious.

Here's a linear solution in which $n$ is a natural variable. We can try to find $n^3$ in terms of $(n-1)^3$. We find
\[
n^3 = (n-1)^3 + 3n^2 - 3n + 1
\]

The problem is the occurrence of $n^2$. But maybe we can find it the same way, in terms of $(n-1)^2$ using the identity
\[
n^2 = (n-1)^2 + 2n - 1
So we need a variable $x$ for the cubes and a variable $y$ for the squares.

\[ x' = n^3 \iff x' = n^3 \land y' = n^2 \]
\[ x' = n^3 \land y' = n^2 \iff \]
if $n=0$ then $x:=0$, $y:=0$ else $n:=n-1$. $x' = n^3 \land y' = n^2$.

We cannot complete that refinement due to a little problem: in order to get the new values of $x$ and $y$, we need not only the values of $x$ and $y$ just produced by the recursive call, but also the original value of $n$, which was not saved. So we revise.

\[ x' = n^3 \iff x' = n^3 \land y' = n^2 \land n'=n \]
\[ x' = n^3 \land y' = n^2 \land n'=n \iff \]
if $n=0$ then $x:=0$, $y:=0$
else $n:=n-1$. $x' = n^3 \land y' = n^2 \land n'=n$. $n:=n+1$.
\[ y:=y+n+n-1. \quad x:=x+y+y-n-n-n+1 \]

After we decrease $n$, the recursive call promises to leave it alone, and then we increase it back to its original value, which fulfills the promise. With recursive time,

\[ x' = n^3 \land t'=t+n \iff x' = n^3 \land y' = n^2 \land n'=n \land t'=t+n \]
\[ x' = n^3 \land y' = n^2 \land n'=n \land t'=t+n \iff \]
if $n=0$ then $x:=0$, $y:=0$
else $n:=n-1$. $t:=t+1$. $x' = n^3 \land y' = n^2 \land n'=n \land t'=t+n$. $n:=n+1$.
\[ y:=y+n+n-1. \quad x:=x+y+y-n-n-n+1 \]

The proof is easier if we express the specifications in program form:

\[ x:=n^3. \quad t:=t+n \iff x:=n^3. \quad y:=n^2. \quad t:=t+n \]
\[ x:=n^3. \quad y:=n^2. \quad t:=t+n \iff \]
if $n=0$ then $x:=0$, $y:=0$
else $n:=n-1$. $t:=t+1$. $x:=n^3$. $y:=n^2$. $t:=t+n$. $n:=n+1$.
\[ y:=y+n+n-1. \quad x:=x+y+y-n-n-n+1 \]

Now we can use the substitution law more.

Here's another linear solution. It is similar to the previous solution, calculating $n^3$ from $(n-1)^3$. The recursion in the previous solution requires a stack implementation; the recursion in this solution does not require a stack implementation. This solution uses a backward-looking specification. Let $n$ be a natural constant, and let $x$ be a natural variable. The result we want is

\[ R \iff x'=n^3 \land t'=t+n \]

We want that result by a sequence of additions to $x$. Let $k$ be a natural variable that counts up from 0 to $n$. Define

\[ Q \iff x=k^3 \implies x'=n^3 \land t'=t+n-k \]

to say that, in the middle of the computation, we have already computed $x=k^3$, and we need to finish computing $x'=n^3$ in time $n-k$. Then

\[ R \iff k:=0. \quad x:=0. \quad Q \]
\[ Q \iff \text{if } k=n \text{ then ok else } x:=x+y. \quad k:=k+1. \quad t:=t+1. \quad Q \text{ fi} \]

where $y$ is a value yet to be determined. The proof of the $R$ refinement is two uses of the substitution law. The proof of the $Q$ refinement is two cases. The first case $k=n$ is easy. The other case $k<n$ is

\[ Q \iff k<n \land (x:=x+y. \quad k:=k+1. \quad t:=t+1. \quad Q) \quad \text{expand second } Q \]
\[ = Q \iff k<n \land (x:=x+y. \quad k:=k+1. \quad t:=t+1. \quad x=k^3 \implies x'=n^3 \land t'=t+n-k) \quad \text{substitution 3 times} \]
\[ = Q \iff k<n \land (x+y=(k+1)^3 \implies x'=n^3 \land t'=t+1+n-(k+1)) \quad \text{simplify} \]
\[ = Q \iff k<n \land (x+y=k^3+3\times k^2+3\times k+1 \implies x'=n^3 \land t'=t+n-k) \quad \text{mirror and expand } Q \]
\[ \iff k<n \land (x+y=k^3+3\times k^2+3\times k+1 \implies x'=n^3 \land t'=t+n-k) \]
\[ \iff x=k^3 \implies x'=n^3 \land t'=t+n-k \]

If we somehow had $y=3\times k^2+3\times k+1$, then by specialization
haven't got many operations to work with. We can try to accumulate a sum, as follows.

Here's the same linear solution using a forward-looking

\[
\begin{align*}
Q & \Leftarrow k=0. \ x:=0. \ y:=1. \ Q \\
R & \Leftarrow \begin{cases}
\text{if } k=n \text{ then } & x:=x+y. \ y:=y+z. \ k:=k+1. \ t:=t+1. \ Q \text{ fi} \\
\text{else } & x:=x+y. \ y:=y+z. \ k:=k+1. \ t:=t+1. \ Q \text{ fi}
\end{cases}
\end{align*}
\]
where \( z \) is a value yet to be determined. The proof of the \( R \) refinement is three uses of the substitution law. The proof of the \( Q \) refinement is two cases. The first case \( k=n \) is easy. The other case \( k<n \) is

\[
\begin{align*}
Q & \Leftarrow k<n \land (x:=x+y. \ y:=y+z. \ k:=k+1. \ t:=t+1. \ Q) \quad \text{expand second } Q \\
& \Rightarrow k<n \land (x:=x+y. \ y:=y+z. \ k:=k+1. \ t:=t+1. \\
& \quad \land x=k^3 \land y=3xk^2+3xk+1 \Rightarrow x'=n^3 \land t'=t+n-k)
\end{align*}
\]
substitution 4 times

\[
\begin{align*}
Q & \Leftarrow k<n \land (x+y=k^3+3xk^2+3xk+1 \land y+z=3xk^2+9xk+7) \\
& \Rightarrow x'=n^3 \land t'=t+n-k
\end{align*}
\]
mirror and expand \( Q \)

\[
\begin{align*}
& \Rightarrow (x=k^3 \land y=3xk^2+3xk+1 \Rightarrow x'=n^3 \land t'=t+n-k) \\
& \text{If we somehow had } z=6xk+6, \text{ then by specialization}
\end{align*}
\]

So we see what \( z \) has to be. Let's just give it to ourselves by modifying \( Q \).

\[
\begin{align*}
Q & \Leftarrow x=k^3 \land y=3xk^2+3xk+1 \land z=6xk+6 \Rightarrow x'=n^3 \land t'=t+n-k
\end{align*}
\]
Now we need to modify our refinements to initialize and update natural variable \( z \).

\[
\begin{align*}
R & \Leftarrow k:=n. \ x:=0. \ y:=1. \ z:=6. \ Q \\
Q & \Leftarrow \begin{cases}
\text{if } k=0 \text{ then } & x:=x+y. \ y:=y+z. \ z:=z+w. \ k:=k+1. \ t:=t+1. \ Q \text{ fi} \\
\text{else } & x:=x+y. \ y:=y+z. \ k:=k+1. \ t:=t+1. \ Q \text{ fi}
\end{cases}
\end{align*}
\]
where \( w \) is a value yet to be determined. The second case \( k<n \) of the \( Q \) refinement is

\[
\begin{align*}
& \Rightarrow k<n \land (x:=x+y. \ y:=y+z. \ z:=z+w. \ k:=k+1. \ t:=t+1. \ Q) \quad \text{expand second } Q \\
& \text{and use substitution 5 times and simplify}
\end{align*}
\]

\[
\begin{align*}
& \Rightarrow k<n \land (x+y=k^3+3xk^2+3xk+1 \land y+z=3xk^2+9xk+7 \land z+w=6xk+12) \\
& \Rightarrow x'=n^3 \land t'=t+n-k
\end{align*}
\]
mirror and expand \( Q \)

\[
\begin{align*}
& \Rightarrow (x=k^3 \land y=3xk^2+3xk+1 \land z=6xk+6 \Rightarrow x'=n^3 \land t'=t+n-k) \\
& \text{If } w=6, \text{ then by specialization}
\end{align*}
\]

So we see that \( w \) has to be 6. The solution is

\[
\begin{align*}
R & \Leftarrow k:=n. \ x:=0. \ y:=1. \ z:=6. \ Q \\
Q & \Leftarrow \begin{cases}
\text{if } k=0 \text{ then } & x:=x+y. \ y:=y+z. \ z:=z+6. \ k:=k+1. \ t:=t+1. \ Q \text{ fi} \\
\text{else } & n:=n-1. \ t:=t+1. \ Q. \ x:=x+y. \ y:=y+z. \ z:=z+6 \text{ fi}
\end{cases}
\end{align*}
\]

The solution is simple and efficient, and we couldn't have found it without using the theory.

Here's the same linear solution using a forward-looking \( Q \), but the recursion requires a stack. Let

\[
\begin{align*}
Q & \Leftarrow x'=n^3 \land y'=3xn^2+3xn+1 \land z'=6xn+6 \land t'=t+n
\end{align*}
\]
Then

\[
\begin{align*}
x'=n^3 \land t'=t+n & \Leftarrow Q \\
Q & \Leftarrow \begin{cases}
\text{if } n=0 \text{ then } x:=0. \ y:=1. \ z:=6 \\
\text{else } n:=n-1. \ t:=t+1. \ Q \ x:=x+y. \ y:=y+z. \ z:=z+6 \text{ fi}
\end{cases}
\end{align*}
\]
Now here's the same solution using the invariant for-loop rule in Subsection 5.2.3. We haven't got many operations to work with. We can try to accumulate a sum, as follows.
\[ x' = n^3 \iff x := 0. \quad \textbf{for} \ k := 0;..;n \ \textbf{do} \ x := x+? \ \textbf{od} \]

where the question mark means we don't know what goes here yet. We define invariant
\[ A \ k \quad \Rightarrow \quad x = k^3 \]

Then
\[ x' = n^3 \iff x := 0. \quad A \ 0 \Rightarrow A' \ n \]
is easily proven. Now, for free,
\[ A \ 0 \Rightarrow A' \ n \quad \iff \quad \textbf{for} \ k := 0;..;n \ \textbf{do} \ k := 0;..;n \land A \ k \Rightarrow A'(k+1) \ \textbf{od} \]

and what remains is to refine \( k := 0;..;n \land A \ k \Rightarrow A'(k+1) \)
\[ k := 0;..;n \land A \ k \Rightarrow A'(k+1) \quad \text{drop} \ k := 0;..;n \text{ and expand} \ A \ k \text{ and} \ A'(k+1) \]
\[ \iff \quad x = k^3 \Rightarrow x' = (k+1)^3 \quad \text{context} \]
\[ \iff \quad x = k^3 \Rightarrow x' = x^3 + 3x^2k + 3xk + 1 \]
\[ \iff \quad x := x + 3x^2k + 3xk + 1 \]

Unfortunately, we don't have squaring or multiplication. So let's just say \( x := x + y \) and strengthen the invariant \( A \ k \) to
\[ A \ k \quad \Rightarrow \quad x = k^3 \land y = 3x^2k + 3xk + 1 \]

Now we must revise the initialization
\[ x' = n^3 \iff x := 0. \quad y := 1. \quad A \ 0 \Rightarrow A' \ n \]

and recalculate the loop body
\[ k := 0;..;n \land A \ k \Rightarrow A'(k+1) \quad \text{drop} \ k := 0;..;n \text{ and expand} \ A \ k \text{ and} \ A'(k+1) \]
\[ \iff \quad x = k^3 \land y = 3x^2k + 3xk + 1 \Rightarrow x' = (k+1)^3 \land y' = 3x(k+1)^2 + 3x(k+1) + 1 \]
\[ \iff \quad x := x + y. \quad y := y + 3xk + 6 \]
\[ \iff \quad x := x + y. \quad y := y + 3xk + 6 \]

and we're done, but it's a little inelegant to add up \( 6k \) so let's say \( y := y + z \) and strengthen \( A \ k \) again to
\[ A \ k \quad \Rightarrow \quad x = k^3 \land y = 3x^2k + 3xk + 1 \land z = 6xk + 6 \]

Now we must revise the initialization
\[ x' = n^3 \iff x := 0. \quad y := 1. \quad z := 6. \quad A \ 0 \Rightarrow A' \ n \]

and recalculate the loop body
\[ k := 0;..;n \land A \ k \Rightarrow A'(k+1) \]
\[ \iff \quad x = k^3 \land y = 3x^2k + 3xk + 1 \land z = 6xk + 6 \]
\[ \iff \quad x' = (k+1)^3 \land y' = 3x(k+1)^2 + 3x(k+1) + 1 \land z' = 6x(k+1) + 6 \]
\[ \iff \quad x := x + y. \quad y := y + 3xk + 6 \land z := z + 6 \]
\[ \iff \quad x := x + y. \quad y := y + z. \quad z := z + 6 \]

and we're done again. Altogether,
\[ x' = n^3 \iff x := 0. \quad y := 1. \quad z := 6. \quad \textbf{for} \ k := 0;..;n \ \textbf{do} \ x := x + y. \quad y := y + z. \quad z := z + 6 \ \textbf{od} \]