243 (bit sum) Write a program to find the number of ones in the binary representation of a given natural number.

After trying the question, scroll down to the solution.

§ Let $f\,n$ be the number of ones in the binary representation of natural n, defined inductively as follows.

$$f\,0 = 0$$
$$f\,(2{\times}n) = f\,n$$
$$f\,(2{\times}n+1) = f\,n + 1$$

Here's one solution. Let n and c be natural variables.

$$c' = f\,n \;\Leftarrow\; c{:=}\,0.\;\; c' = c + f\,n$$
$$c' = c + f\,n \;\Leftarrow\; \textbf{if } n{=}0 \textbf{ then } ok$$
$$\textbf{else } \textbf{if } even\ n \textbf{ then } n{:=}\,n/2.\;\; c' = c + f\,n$$
$$\textbf{else } n{:=}\,(n{-}1)/2.\;\; c{:=}\,c{+}1.\;\; c' = c + f\,n \textbf{ fi fi}$$

Proof of first refinement:

$c{:=}\,0.\;\; c' = c + f\,n$ — substitution law, arithmetic
$=\;\; c' = f\,n$

The last refinement is proven by cases. First case:

$n{=}0 \wedge ok$ — expand ok
$=\;\; n{=}0 \wedge c'{=}c \wedge n'{=}n$ — $f\,0 = 0$
$=\;\; n{=}0 \wedge c' = c + f\,0 \wedge n'{=}n$ — context from left conjunct to change middle conjunct
$=\;\; n{=}0 \wedge c' = c + f\,n \wedge n'{=}n$ — specialization
$\Rightarrow\;\; c' = c + f\,n$

Middle case:

$n{>}0 \wedge even\ n \wedge (n{:=}\,n/2.\;\; c' = c + f\,n)$ — substitution law
$=\;\; n{>}0 \wedge even\ n \wedge c' = c + f(n/2)$ — property of f for even arguments
$=\;\; n{>}0 \wedge even\ n \wedge c' = c + f\,n$ — specialization
$\Rightarrow\;\; c' = c + f\,n$

Last case:

$odd\ n \wedge (n{:=}\,(n{-}1)/2.\;\; c{:=}\,c{+}1.\;\; c' = c + f\,n)$ — substitution law twice
$=\;\; odd\ n \wedge c' = c{+}1{+}f((n{-}1)/2)$ — property of f for odd arguments
$=\;\; odd\ n \wedge c' = c + f\,n$ — specialization
$\Rightarrow\;\; c' = c + f\,n$

The execution time is exactly

$$\textbf{if } n{=}0 \textbf{ then } 0 \textbf{ else } floor\,(1 + log\ n) \textbf{ fi}$$

or, for easier proof,

$$(n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$$

Proof of first refinement:

$c{:=}\,0.\;\; (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$ — substitution law
$=\;\; (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$

The last refinement is proven by cases. First case:

$n{=}0 \wedge ok$ — expand ok and drop useless conjuncts
$=\;\; n{=}0 \wedge t'{=}t$ — discharge and identity
$=\;\; n{=}0 \wedge (n{=}0 \Rightarrow t'{=}t) \wedge \top$ — use context $n{=}0$
$=\;\; n{=}0 \wedge (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$ — specialization
$\Rightarrow\;\; (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$

Middle case:

$n{>}0 \wedge even\ n \wedge (n{:=}\,n/2.\;\; t{:=}\,t{+}1.\;\; (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\ n))$
— substitution law twice
$=\;\; n{>}0 \wedge even\ n \wedge (n/2{=}0 \Rightarrow t'{=}t{+}1) \wedge (n/2{>}0 \Rightarrow t' \le t + 2 + log\ (n/2))$
— context $n{>}0$ means $n/2{=}0$ is $\bot$ and $n/2{>}0$ is $n{>}0$
$=\;\; n{>}0 \wedge even\ n \wedge \top \wedge (n{>}0 \Rightarrow t' \le t + 2 + log\ (n/2))$
— context $n{>}0$ means $n{=}0$ is $\bot$
$=\;\; n{>}0 \wedge even\ n \wedge (n{=}0 \Rightarrow t'{=}t) \wedge (n{>}0 \Rightarrow t' \le t + 2 + log\ (n/2))$
— $1 + log\ (n/2) \;=\; log\ n$
$=\;\; n{>}0 \wedge even\ n \wedge (n{=}0 \Rightarrow t'{=}t) \wedge (n{>}0 \Rightarrow t' \le t + 1 + log\ n)$ — specialization

$\Longrightarrow \quad (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\; n)$

Last case:

$odd\; n \wedge (n{:=}\; (n-1)/2.\;\; c{:=}\; c+1.\;\; t{:=}\; t+1.\; (n{=}0 \Rightarrow t'{=}t) \wedge (n{>}0 \Rightarrow t' \le t + 1 + log\; n))$

substitution law 3 times

$= \quad odd\; n \wedge ((n-1)/2{=}0 \Rightarrow t'{=}t+1) \wedge ((n-1)/2{>}0 \Rightarrow t' \le t + 2 + log\; ((n-1)/2))$

various simplifications

$= \quad odd\; n \wedge (n{=}1 \Rightarrow t'{=}t+1) \wedge (n{>}1 \Rightarrow t' \le t + 1 + log\; (n-1))$

combine the middle and last conjunct

$\Longrightarrow \quad odd\; n \wedge (n{\ge}1 \Rightarrow t' \le t + 1 + log\; (n-1))$ use context to conjoin $\top$

$= \quad odd\; n \wedge (n{=}0 \Rightarrow t'{=}t) \wedge (n{\ge}1 \Rightarrow t' \le t + 1 + log\; (n-1))$ specialization

$\Longrightarrow \quad (n{=}0 \Rightarrow t'{=}t) \;\wedge\; (n{>}0 \Rightarrow t' \le t + 1 + log\; n)$

Here's another solution. Let n be a natural variable.

$n' = f\,n \;\Longleftarrow\;$ **if** $n{=}0$ **then** ok
else if $even\; n$ **then** $n{:=}\; n/2.\;\; n' = f\,n$
else $n{:=}\; (n-1)/2.\;\; n' = f\,n.\;\; n{:=}\; n+1$ **fi fi**

with the same execution time. First, we prove the result without the execution time. There's only one variable, n , so $n' = f\,n \;=\; n{:=} f\,n$. Therefore

if $n{=}0$ **then** ok
else if $even\; n$ **then** $n{:=}\; n/2.\;\; n' = f\,n$
else $n{:=}\; (n-1)/2.\;\; n' = f\,n.\;\; n{:=}\; n+1$ **fi fi**

$=$ **if** $n{=}0$ **then** $n'{=}n$
else if $even\; n$ **then** $n{:=}\; n/2.\;\; n' = f\,n$
else $n{:=}\; (n-1)/2.\;\; n{:=} f\,n.\;\; n' = n+1$ **fi fi**

context, substitution law three times

$=$ **if** $n{=}0$ **then** $n'{=}0$
else if $even\; n$ **then** $n' = f(n/2)$
else $n' = f(((n-1)/2)+1)$ **fi fi**

Those are the same three cases as the definition of f . (This proof is slightly unfinished. The timing should also be proven.)