252 Let n be a natural variable. Add time according to the recursive measure, and find a finite upper bound on the execution time of

$$P \Leftarrow \textbf{if } n \ge 2 \textbf{ then } n:= n-2.\ P.\ n:= n+1.\ P.\ n:= n+1 \textbf{ else } ok \textbf{ fi}$$

§ To ensure that every loop includes a time increment, it is sufficient to put $t:= t+1$ just before the first call. (But the question isn't any harder, and the time bound isn't significantly different, if we put $t:= t+1$ before both calls.) Because of the two calls, each at approximately the original value of n , I guess the time might be exponential. Actually, it looks just like Fibonacci: the first call is at $n-2$, the second is at $n-1$. Let's try

$$P = t' \le t + 2^n$$

The proof of the refinement will be by cases. First case:

$$n{\ge}2 \wedge (n:= n-2.\ t:= t+1.\ P.\ n:= n+1.\ P.\ n:= n+1)$$
$$= \quad n{\ge}2 \wedge (t' \le t + 1 + 2^{n-2}.\ t' \le t + 2^{n+1}.\ n'=n+1 \wedge t'=t)$$
$$= \quad n{\ge}2 \wedge \exists n'', t'', n''', t'''\cdot\ t'' \le t + 1 + 2^{n-2} \wedge t''' \le t'' + 2^{n''+1} \wedge n'=n'''+1 \wedge t'=t'''$$
$$= \quad n{\ge}2$$

Oops. The final time seems to be completely arbitrary. The problem is that the first call of P allows n to change arbitrarily, so the last call of P allows t to change arbitrarily. Let's try again.

$$P = n'=n \wedge t' \le t + 2^n$$

The proof of the refinement will be by cases. First case:

$$n{\ge}2 \wedge (n:= n-2.\ t:= t+1.\ P.\ n:= n+1.\ P.\ n:= n+1)$$
$$= \quad n{\ge}2 \wedge n'=n \wedge t' \le t + 1 + 2^{n-2} + 2^{n-1}$$
$$= \quad n{\ge}2 \wedge n'=n \wedge t' \le t + 1 + 3{\times}2^{n-2} \qquad \text{when } n{\ge}2 ,\ 1 \le 2^{n-2}$$
$$\Rightarrow \quad n{\ge}2 \wedge n'=n \wedge t' \le t + 2^{n-2} + 3{\times}2^{n-2} \qquad \text{specialize and simplify}$$
$$\Rightarrow \quad n'=n \wedge t' \le t + 2^n$$

Last case:

$$n{<}2 \wedge ok$$
$$= \quad n{<}2 \wedge n'=n \wedge t'=t \qquad \text{and } 0 \le 2^n$$
$$\Rightarrow \quad n'=n \wedge t' \le t + 2^n$$