Problem for 2006 September

Proposed by Dan Jurca

The following program written in C/C++ can be used to (recursively) compute the Fibonacci numbers \( F_n \), where \( F_0=0, F_1=1, \) and \( 2 \leq n \Rightarrow F_n=F_{n-2}+F_{n-1} \).

```c
unsigned int fib(unsigned int n)
{
    return( n < 2 ? n : fib(n-2) + fib(n-1) );
}
```

Let \( C_n \) be the number of times this function is entered when it is used to compute \( F_n \). For example, \( C_0=C_1=1 \) and \( C_2=3 \). Determine \( C_n \) as a function of \( n \).

Solution by the proposer

Proposition. \( 0 \leq n \Rightarrow C_n=2F_{n+1}-1. \)

Proof.

We have

\[
C_0 = 1 = 2 \cdot 1 - 1 = 2F_1 - 1 = 2F_{0+1}-1 \quad \text{and} \\
C_1 = 1 = 2 \cdot 1 - 1 = 2F_2 - 1 = 2F_{1+1}-1,
\]

so the proposition certainly holds if \( n=0 \) or \( n=1 \). Suppose now that \( 2 \leq n, C_{n-2}=2F_{n-1}-1, \) and \( C_{n-1}=2F_n-1 \). Then since, obviously,

\[
C_n = 1 + C_{n-2} + C_{n-1}, \quad \text{we have} \\
C_n = 1 + (2F_{n-1}-1) + (2F_n-1) \\
= 2(F_{n-1} + F_n) - 1 \\
= 2F_{n+1}-1,
\]
so the proposition follows by induction on $n$.

Also solved by Massoud Malek and John M. Sayer