A constant-time solution to the staircase problem

How many distinct ways are there to climb an n step staircase by taking only 1 or 2 steps at a time? This common programming interview question is typically solved via recursive ($\mathcal{O}(2^n)$) or iterative ($\mathcal{O}(n)$) procedures. However, there is also a (mostly impractical) solution that runs in constant-time. A staircase consisting of just one step can only be climbed by taking one 1-step. A staircase consisting of two steps can be climbed by taking two 1-steps or one 2-step. So $a_1 = 1, a_2 = 2$. Let $k \geq 3$. The final move climbing a staircase of k steps can be a 1-step from a staircase of $k-1$ steps or a 2-step from a staircase of $k-2$ steps. So $a_k = a_{k-1} + a_{k-2}$. Let $n \in \mathbb{N}, t \in \mathbb{R}$. If r is a root of $t^2 - t - 1$ then $r^2 = r + 1$. Applying the quadratic formula, the roots of $t^2 - t - 1$ are $\frac{-(-1)\pm\sqrt{(-1)^2+4(1)(-1)}}{2(1)} = \frac{1\pm\sqrt{5}}{2}$. Let $r_1 = \frac{1-\sqrt{5}}{2}, r_2 = \frac{1+\sqrt{5}}{2}$. Let $x, y \in \mathbb{R}$. Then

$$\begin{aligned} xr_1^{n-1} + yr_2^{n-1} + xr_1^{n-2} + yr_2^{n-2} &= xr_1^{n-2}(r_1 + 1) + yr_2^{n-2}(r_2 + 1) \\ &= xr_1^{n-2}r_1^2 + yr_2^{n-2}r_2^2 \\ &= xr_1^n + yr_2^n \end{aligned}$$

So $xr_1^n + yr_2^n$ satisfies the recursion. Solving the system of equations $1 = ar_1^1 + br_2^1$ and $2 = ar_1^2 + br_2^2$ given by our base cases, we get the real numbers $a = \frac{5-\sqrt{5}}{10}, b = \frac{5+\sqrt{5}}{10}$. So

$$\left(\frac{5-\sqrt{5}}{10}\right)\left(\frac{1-\sqrt{5}}{2}\right)^n + \left(\frac{5+\sqrt{5}}{10}\right)\left(\frac{1+\sqrt{5}}{2}\right)^n$$

satisfies the base cases $a_1 = 1, a_2 = 2$ and the recursion $a_n = a_{n-1} + a_{n-2}$. All of the math performed in this solution maps directly to constant-time machine code instructions for floating-point numbers. However, very high precision floating-point numbers or symbolic evaluation is required for even remotely high values of n. Still, it's surprising how often this solution (that anyone who has taken discrete math should be able to conjure up) goes unmentioned when discussing the problem in a programming context.

```
#include <iostream>
#include <ginac/ginac.h>

using namespace std;
using namespace GiNaC;

ex a = numeric(1, 10) * (5 - sqrt(ex(5)));
ex b = numeric(1, 10) * (5 + sqrt(ex(5)));
ex r1 = (1 - sqrt(ex(5))) / 2;
ex r2 = (1 + sqrt(ex(5))) / 2;

ex staircase_closed_form(uint n)
{
    ex result = (a * pow(r1, n) + b * pow(r2, n));
    return result;
}

int main()
{
    cout << staircase_closed_form(1).normal() << endl; // 1
    cout << staircase_closed_form(2).normal() << endl; // 2
    cout << staircase_closed_form(3).normal() << endl; // 3
    cout << staircase_closed_form(100).normal() << endl; // 573147844013817084101
}
```

Figure 1: Symbolic implementation with GiNaC