CPSC 303 Will

Fibonacci Numbers

$F_0 = 0 \quad F_1 = 1$

homogeneous $\longrightarrow$ $F_n = F_{n-1} + F_{n-2}$ for $n \geq 2$

We know: $F_n = \left[\frac{\phi^n}{\sqrt{5}} \right]$ ← round to nearest integer

$\phi = \frac{1+\sqrt{5}}{2}$

$= 1.618\ldots$

Alg 1

```
int fib(n)
    if n==0 return 0
    if n==1 return 1
    return fib(n-1) + fib(n-2)
```

Correctness? YES

Runtime: $T(n)$ is # lines of code executed by fib(n)

$T(0) = 1 \quad T(1) = 2$

inhomogeneous $\longrightarrow$ $T(n) = 3 + T(n-1) + T(n-2)$

$> F_n \approx (1.6)^n$

Is this fast? NO exponential running time

Alg 2

```
int fib(n)
    F[0] = 0
    F[1] = 1
    for i = 2 to n
        F[i] = F[i-1] + F[i-2]
    return F[n].
```

executed $n-2$ times

costly for large integers

Correctness? YES

Runtime? $\Theta(n)$

To calculate F_n, numbers that are added together are n-bit numbers

$\log_2(\varphi^n) \simeq n \log_2(1.6)$

Time to add two n-bit numbers is ~~$O(n)$~~

$O(n^2)$ using big number arithmetic

Alg 3 Use the formula !!!

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

$\sqrt{5}$ is irrational.

What precision do we need to calculate F_n correctly?

I DON'T KNOW.

Alg 4 Linear Algebra

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

$$\begin{pmatrix} F_2 \\ F_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} F_0 \\ F_1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^2\begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

"A"

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = A^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

fib(n)

1. Compute $\boxed{A^n}$
2. $\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = A^n \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$
3. return F_n

Repeated Squaring:

$\boxed{A^{19}} = A^{16} \cdot A^2 \cdot A$

$A \cdot A = A^2$

$A^2 \cdot A^2 = A^4$

$A^4 \cdot A^4 = A^8$

$A^8 \cdot A^8 = A^{16}$

$O(\log n)$ matrix multiplications to get A^n

↑ time to multiple two n-bit numbers.

in 2019 D. Harvey and J. van der Hoeven showed can be done in $O(n \log n)$

$O(n \log n)$ for this Alg 4