Computing Fibonacci Nos.

i.e. given \( n \), compute \( F_n \)

where \( F_0 = 0 \) \( \quad F_1 = 1 \) \( \quad F_2 = F_1 + F_0 \) \( \quad \text{otherwise} \)

Method I: write the equivalent recursive program

Method II

\[
\begin{align*}
F_2 &= 0 + 1 = 1 \\
F_3 &= F_2 + F_1 \\
\vdots & \ldots \\
F_n &= \quad \text{(expression)}
\end{align*}
\]

Time for Method II

\( T^{II}(n) \): the no. of steps (instructions executed)

for computing \( F_n \) using Method II

About \( \frac{n}{2} \) iterations where in each iteration we sum two previously computed Fib nos.

\( \Rightarrow O(n) \) additions Space: 2 nos.
\[ F_n = F_{n-1} + F_{n-2} \]

\[ T^I(n) = T^I(n-1) + T^I(n-2) + 1 \]

- To compute \( F_{n-1} \) recursively.
- \( T(1) = 1 \), \( T(0) = 1 \)

\[ T^I(n) = ? \text{ at least } F_n \text{ which is roughly } (1.6)^n \text{ additions} \]

Method I
Cost of addition?

In Method II, if you consider the last $\frac{n}{2}$ iterations, we are adding no. of size $\frac{n}{2}$ bits. Adding two $b$ bit nos. takes $O(b)$ steps.

The last $\frac{n}{2}$ iteration cost $O(n)$ steps.

$\Omega(n^2)$ steps overall.

What is the min time to compute $F_n$?

Any algorithm must take time $\Omega$ (input size + output size).

$\Omega(n)$ is a lower bound for $F_n$. 
\[
\begin{bmatrix}
F_i \\
F_{i-1}
\end{bmatrix} = 
\begin{bmatrix}
1 & 1 \\
1 & 0
\end{bmatrix}
\begin{bmatrix}
F_{i-1} \\
F_{i-2}
\end{bmatrix}
\]

\[
A^2 = 
\begin{bmatrix}
1 & 1 \\
1 & 0
\end{bmatrix} 
\begin{bmatrix}
1 & 1 \\
1 & 0
\end{bmatrix} 
\begin{bmatrix}
F_{i-2} \\
F_{i-3}
\end{bmatrix}
\]

\[
F_n = \left(A^n\right)_{1,1} \\
\]

What is the time to compute \(X^n\)

\[
X^n = \begin{cases} 
(x^{\frac{n}{2}})^2 & \text{if } n \text{ even} \\
X \cdot (X^{\frac{n-1}{2}}) & \text{otherwise}
\end{cases}
\]

\[
\log n \text{ multiplications; but what is the size?}
\]

\[
|X^n| = n \log_2 x
\]