Chapter 0

Notes: some of the answers are gathered/referenced from online resources, contact me if you think some of the solutions are your work and should be cited.

0.1 (a) Θ (b) O (c) Θ (d) Θ
(e) Θ (f) Θ (g) Ω (h) Ω (i) Ω
(j) Ω (k) Ω (l) O (m) O (n) Θ
(o) Ω (p) O (q) Θ

* (g) ~ Use L'Hôpital's rule

** for (p), Consider turn power functions into $n^{f(n)}$ form (hint, factoring on both sides)

0.2 (a) $\because c < 1$

$\therefore \frac{1 + c + c^2 + \cdots + c^n}{1} = \frac{\frac{1}{1-c}}{1} \leq k$, k being a positive integer, so $g(n) = O(1)$, and $\frac{1}{g(n)} = 1$ if $c \to 0$, so $g(n) = \Theta(1)$

(b) $1 + c + c^2 + \cdots + c^n = n$ if $c = 1$

$g(n) = \Theta(n)$

(c) follow "the moral", when $c > 1$ $g(n)$ if non-decreasing, $g(n) = O(c^n) = \Theta(c^n)$

0.3 (a) Basis : $n=6$, this holds

Suppose : $F_n \geq 2^{\frac{n}{2}}$, for $n \geq 6$

Prove : $F_{n+1} \geq 2^{\frac{n+1}{2}}$

$$F_{n+1} = F_n + F_{n-1} \geq 2^{\frac{n}{2}} + 2^{\frac{n-1}{2}}$$

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

$$= 2^{\frac{n}{2}} \cdot \frac{\sqrt{2}+1}{\sqrt{2}} > 2^{\frac{n}{2}} \cdot \frac{2}{\sqrt{2}}$$

$$= 2^{\frac{n+1}{2}}$$

(b) $\lim_{n\to\infty} F_n = \frac{\varphi^n - (-\frac{1}{\varphi})^n}{\sqrt{5}}$, where $\varphi = \frac{1+\sqrt{5}}{2}$

So, $F(n) = \Theta(\varphi^n)$

to find c, $2^{cn} = \left(\frac{1+\sqrt{5}}{2}\right)^n$,

implies $c = \log_2(1+\sqrt{5}) - 1$

Added explanations, $\varphi^2 = \varphi + 1$, so $F_n \le \varphi^n$ and $F_{n+1} \le \varphi^{n+1}$ implies:

$$F_{n+2} = F_n + F_{n+1} \le \varphi^{n+1} + \varphi^n = (\varphi+1)\varphi^n$$
$$= \varphi^2 \varphi^n = \varphi^{n+2}$$

$F_0 \le 1 \le \varphi^0$, $F_1 \le 1 < \varphi^1$, so,
$F_n \le \varphi^n$ for all $n \ge 0$.

Let $c = \log_2 \varphi$.

(c) Let $b = 2^c$, $F_{n+2} = F_n + F_{n+1}$, $b^{n+2} = b^n + b^{n+1}$ which is $b^{n+2} - b^n - b^{n+1} = 0$, solving for it is $b = \frac{1 \pm 5^{\frac{1}{2}}}{2}$, then, $c = \log_2 b = \log_2 \left(\frac{1+5^{\frac{1}{2}}}{2}\right)$

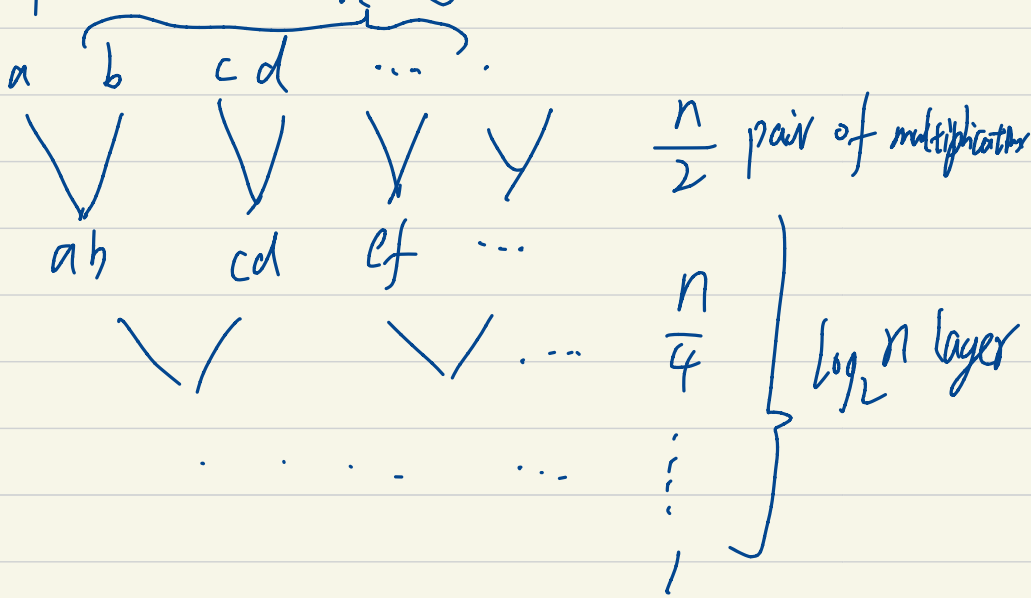
≈ 0.69

0.4 (a) For the former half:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{pmatrix}$$

4 additions, 8 multiplications.

For the later half: we are multiplication in an order of a reversed-binary tree like manner



So, in total, $\frac{n}{2} + \frac{n}{4} + \cdots 1 = n\left(1+\frac{1}{2}+\frac{1}{4}+\cdots\right)$
$= 2n$ pair of

bi-matrix multiplication happened.

That is $2n \times 8 = 16n$ times of multiplications.

(b) we know from (a) to know there are $\log_2 n$ layer of multiplications that happens, so $O(\log n)$ would suffice.

More formally, it can be summarized in a function

$$X^n = \begin{cases} I, & \text{if } n=0 \\ X, & \text{if } n=1 \\ X^{n/2} \cdot X^{n/2} & \text{if } n \text{ is even} \\ X^{\lfloor n/2 \rfloor} \cdot X^{\lfloor n/2 \rfloor} & \text{if } n \text{ is odd.} \end{cases}$$

It's a recursive one, and have $\log_2(n)$ layers.

(c) Show that all intermediate results of fib3 are $O(n)$ bits long.

We can use induction to prove the claim. Let

$$F = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

Our base cases compute F^0 and F^1, which are $O(1)$ bits long. Now, assume that the claim holds for $1 \leq n < k$. If k is odd, then $F^k = F \cdot F^{\lfloor k/2 \rfloor} \cdot F^{\lfloor k/2 \rfloor}$. By the inductive hypothesis, computing this requires multiplying $O(1)$ bit numbers with $O(k/2)$ bit numbers and then with $O(k/2)$ bit numbers, resulting in $O(k)$ bit numbers. If k is even, then $F^k = F^{\lfloor k/2 \rfloor} \cdot F^{\lfloor k/2 \rfloor}$, which similarly has $O(k)$ bit numbers. Therefore, the claim holds by induction.

(d) Let $M(n)$ be the running time of an algorithm for multiplying n-bit numbers, and assume that $M(n) = O(n^2)$. Prove that the running time of fib3 is $O(M(n) \log n)$.

fib3 requires $O(\log n)$ multiplications, which each run in $O(M(n))$ time. Each of the $O(\log n)$ steps of the algorithm does $O(1)$ other work, so the algorithm requires $O(M(n) \log n)$ time.

(e) Can you prove that the running time of fib3 is $O(M(n))$?

We can prove this using induction. Our base cases are once again F^0 and F^1, which clearly require $O(M(n))$ time. Now, assume that the claim holds for $1 \leq n < k$. If k is odd, then $F^k = F \cdot F^{\lfloor k/2 \rfloor} \cdot F^{\lfloor k/2 \rfloor}$. By the inductive hypothesis, determining $F^{\lfloor k/2 \rfloor}$ requires $O(M(k/2) = O(M(k))$ time. Multiplying these arrays requires $O(M(k))$ time as well, so F^k can be found in $O(M(k))$ time. Similarly, F^k can be computed in $O(M(k)$ time if k is even. Therefore, the claim holds by induction.