HW2

1. (a) $t_{1000} = 3ns = k \cdot [\log(1000)]^2 \Rightarrow k = \frac{3ns}{\log^2(1000)}$

$$t_{5000} = k \cdot \log^2(5000) = \frac{3ns}{\log^2(1000)} \cdot \log^2(5000)$$

$$= 3ns\left(\frac{\log^2(5000)}{\log^2(1000)}\right)$$

$$\approx \boxed{4.5608\ ns}$$

(b) $t_N = k \cdot \sqrt{N}$, $n = \log_2(N)$, $N = 2^n$

$= k \cdot \sqrt{2^n}$

$t_{N'} = k \cdot \sqrt{N'}$

$= k \cdot \sqrt{2^{n'}}$ $(n' = n+10)$

$= k \cdot \sqrt{2^{n+10}}$

$= k \cdot \sqrt{2^n \cdot 2^{10}}$

$= k \cdot (2^5 \sqrt{2^n})$

$= 2^5 \cdot k\sqrt{2^n}$

$= 32 \cdot t_N$

$= 32 \cdot 11$

$= \boxed{352\ ns}$

2. for RSA decryption, we take $c^d \equiv m \pmod N$

and for modular exponentiation, it takes $O(n^3) = O(\log^3 N)$

if $t_{1024} = k \cdot \log^3 N = k \cdot n^3 = k \cdot 10^3$

$t_{4096} = k \cdot \log^3 N'$

$= k \cdot (n')^3$

$= k \cdot (12)^3$

$= k \cdot 10^3 \cdot \frac{12^3}{10^3}$

$= \boxed{1.728\, t_{1024}}$

So for 4096-bit RSA, it will take 1.728 times as much as the time for decryption of 1024-bit RSA,

3. (a) largest number we have to add is the last addition where we add $\frac{N(N-1)}{2}$ and N

N has $\log(N) = n$ bits

then $\frac{N(N-1)}{2}$ has $2n$ bits

then the running time is $O(2n)$ or $O(2\log N)$

as we have $N-1$ addition, approximately N,

total running time $= N \cdot O(2\log N)$

$\Rightarrow \boxed{O(N \log N)}$

(b) it's like a multiplication of N and $\frac{N+1}{2}$

if N has $n = \log N$ digits

$\frac{N+1}{2}$ has the same

multiplication of $2n$ bits is $O(n^2)$

so overall running time is $\boxed{O(\log^2 N)}$

4. similary, largest number for multiplication is $6^{N-1} \cdot 6$

6 has 3 digit (110)

6^{N-1} has $\log_2(6^{N-1}) = (N-1)\log_2(6)$ digits

So its multiplication takes

$$O(3 \cdot (N-1)\log_2(6))$$

we have $N-1$ multiplication

so in total we have

$$(N-1) \cdot O(3 \cdot (N-1)\log_2(6))$$

$$= O(3\log_2 6 (N-1)^2) \in \boxed{O(N^2)}$$

not sure on this

5. list of F_i : F_1 F_2 F_3 F_4 ... F_{n-1} F_n

approximation in terms of a : a^1 a^2 a^3 a^4 ... a^{n-1} a^n

approximation of # of digits : $\log(a^1)$ $\log(a^2)$ $\log(a^3)$ $\log(a^4)$ $\log(a^{n-1})$ $\log(a^n)$

in binary

↓

simplify : $1\cdot\log a$ $2\cdot\log a$ $3\cdot\log a$ $4\cdot\log a$ $(n-1)\log a$ $n\cdot\log a$

note: we actually have n-1 multiplication, but since the last one involves the longest digits, we can consider that as the worst-case and times the # of multiplication we have.

time for the last multiplication:

as timing a n-digit number and a m-digit numbe will give at most (n+m)-digit number

$\prod_{i=1}^{n-1} F_i \approx \prod_{i=1}^{n-1} a^2$ gives a $\sum_{i=1}^{n-1} i\cdot\log a = \frac{(\log a)n(n-1)}{2}$ digit number

then multiply these two use $O\left(\frac{(\log a)n(n-1)}{2}\cdot n\log a\right)$

$= O\left((\log^2 a)\frac{n^2(n-1)}{2}\right)$

then runtime of all multiplication takes :

$(n-1)\cdot O\left(\log^2 a\,\frac{n^2(n-1)}{2}\right) = O\left(\log^2 a\,\frac{n^2(n-1)^2}{2}\right) \in \boxed{O(n^4)}$