1.$\Theta(n^2)$ $c_1 = 2, c_2 = 3, n_0 = 1$

2.$O(n^2 2^n)$ $c = 101, n_0 = 10$

3.

(1) $O(n^3)$

i will iterate from 0~n-1, it take n times

j will iterate from 0~n-1 each i, it take n^2 times

k will iterate from 0~n-1 each j, it take n^3 times

total cost $n^3 + n^2 + n$

when $c = 2, n_0 = 2$

$n^3 + n^2 + n \leq cn^3, when\ n \geq n_0$

So the time complexity is $O(n^3)$

(2) $O(n^3)$

i will iterate from 0~n-1, it take n times

j will iterate from 0~n-1 each i, it take $\sum_{t=1}^{n} t = \frac{(n+1)n}{2}$ times

k will iterate from 0~n-1 each j, it take $\sum_{t=1}^{n} \frac{(t+1)t}{2} = \frac{\sum_{t=1}^{n} t^2+t}{2} = \frac{2n^3+6n^2+2n}{12}$ times

total cost$\frac{1}{6}n^3 + n^2 + \frac{10}{6}n$

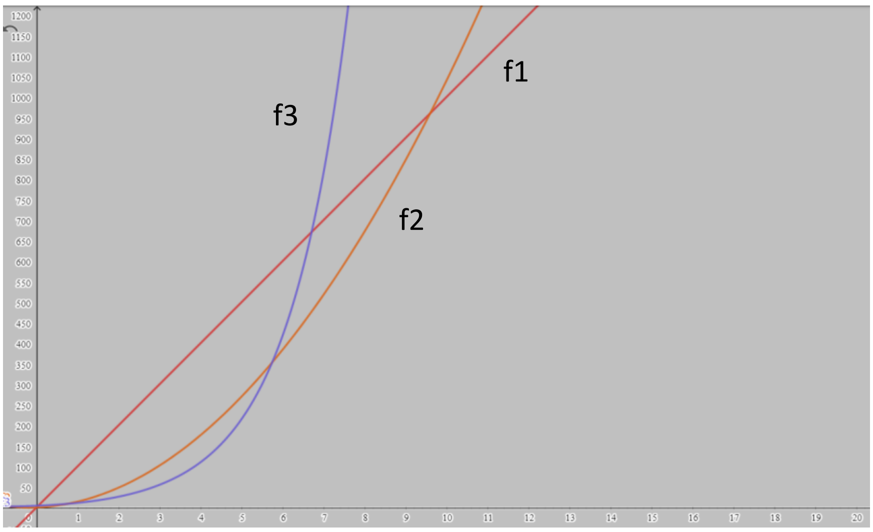
when $c = \frac{2}{6}, n_0 = 8$

$\frac{1}{6}n^3 + n^2 + \frac{10}{6}n \leq cn^3, when\ n \geq n_0$

So the time complexity is $O(n^3)$

4.

(1)



(2) f3'(n)>f2'(n)>f1'(n), when n>=4.8

O(f1(n))=O(n)

O(f2(n))=O(n^2)

O(f3(n))=O(2^n)

(3) O(n^t) is meaning $f(n) \leq c * n^t, when\ n \leq n_0\ f(n) is\ n - polynomial,$

so the constant can put into c

(4) $n^2 + n < cn^2$, when $n \leq n_0$

let c = 2, $n_0 = 0$, then $n^2 + n$'s time complexity is O(n^2)