chapter2

1. prove:

a. because $k \geq d$, so there exist $c = \sum c_i$, to let

$$p(n) = \sum_{n=0}^{d} a_i * n^i \geq \sum_{n=0}^{d} c_i * n^d = c * n^d$$

so there exist c and n_0 to make all the $n \geq n_0$, to make $p(n) \leq cn^k$, so $p(n) = O(n^k)$

b. because $k \leq d$, so there exist c and $n \geq n_0$ to let

$$p(n) = \sum_{n=0}^{d-1} a_i * n^i + a_d * n^d \geq cn^k$$

so there exist c and n_0 to make all the $n \geq n_0$, to make $p(n) \geq cn^k$, so $p(n) = \Omega(n^k)$

c. because $k = d$, so there exist c_1 , c_2 and $n \geq n_0$ to let

$$c_1 n^k \leq p(n) = \sum_{n=0}^{d-1} a_i * n^i + a^k n^k \leq c_2 n^k$$

therefore $p(n) = \Theta(n^k)$

d. because k>d, so there exist $c = \sum c_i$ to let

$$p(n) = \sum_{n=0}^{d} a_i * n^i < \sum_{n=0}^{d} c_i * n^d = c * n^d$$

so there exist c and n_0 to make all the $n \geq n_0$, to make $p(n) < cn^k$, so $p(n) = o(n^k)$

e. because k<d, so there exist c and $n > n_0$ to let

$$p(n) = \sum_{n=0}^{d-1} a_i * n^i + a_d * n^d > cn^k$$

so, $p(n) = \omega(n^k)$

2. show that the solution of $T(n) = T(\lceil n/2 \rceil) + 1$ is O(lgn)

according to master theorem, $a = 1, b = 2, \log_b a = 0, f(n) = O(1) = \Theta(log_b a),$ so $T(n) = \Theta(n^{log_b a} lgn) = \Theta(lgn) = O(lgn)$.

3. Argue that the solution to the recurrence $T(n) = T(n/3) + T(2n/3) + cn$, where c is a constant , is $\Omega(nlogn)$ by appealing to a recursion tree.

as is illstruasted on the picture, the height of tree is $\Omega(lgn)$, each high will cost time cn, so the total time complexity is $\Omega(nlogn)$

4.Use the master method to give tight asymptotic bounds for the following recurrences

a. $T(n) = 4T(n/2) + n$

$a = 4, b = 2, f(n) = n = O(n^{log_b a})$ so there exist a $\epsilon = 1$, so $T(n) = O(n^2)$

b. $T(n) = 4T(n/2) + n^2$

$a = 4, b = 2, f(n) = n^2 = \Theta(n^{log_b a})$, so $T(n) = O(n^2 logn)$

c. $T(n) = 4T(n/2) + n^3$

$a = 4, b = 2, f(n) = n^3 = \Omega(n^{log_b a})$, to make $af(n/b) \leq cf(n)$

which is $4 * (n/2)^3 \leq c * n^3$

$c \geq 0.5$

so , there exist $0.5 \leq c \leq 1$, $T(n) = \Theta(n^3)$

chapter 3

1. prove that couting sort is stable

suppose i<j and A[i]=A[j], if we first get A[j], then C[A[j]]--, next time when we get A[i], we will put A[i] in front of A[j], so couting sort is stable.

2. show how to sort n integers in the range 0 to $n^2 - 1$ in O(n) time .

we can use radix sort. Assuming that the biggest number has d bits, and to sort a bit will take $O(n)$, so the total time complexity is $O(dn) = O(n)$.

3. What is the worst-case running time for the bucket sort? What simple change to the algorithm preserves its linear expected running time and makes its worst-case running time $O(nlogn)$

the worst-case is $O(n^2)$ when all the number is in one bucket. We can change the insert sort to quick sort to make the worst-case become $O(nlogn)$