2.3

Section 2.2 describes a method for solving recurrence relations which is based on analyzing the recursion tree and deriving a formula for the work done at each level. Another (closely related) method is to expand out the recurrence a few times, until a pattern emerges. For instance, let's start with the familiar $T(n) = 2T(n/2) + O(n)$. Think of $O(n)$ as being $\leq cn$ for some constant c, so: $T(n) \leq 2T(n/2) + cn$. By repeatedly applying this rule, we can bound $T(n)$ in terms of $T(n/2)$, then $T(n/4)$, then $T(n/8)$, and so on, at each step getting closer to the value of $T(\cdot)$ we do know, namely $T(1) = O(1)$.

$T(n) \leq 2T(n/2)+cn \leq 2[2T(n/4)+cn/2]+cn = 4T(n/4)+2cn \leq 4[2T(n/8)+cn/4]+2cn = 8T(n/8)+3cn \leq 8[2T(n/16) + cn/8] + 3cn = 16T(n/16) + 4cn$. A pattern is emerging... the general term is $T(n) \leq 2kT(n/2k) + kcn$. Plugging in $k = \log2\ n$, we get $T(n) \leq nT(1) + cn \log2\ n = O(n \log n)$.

(a) Do the same thing for the recurrence T (n) = 3T (n/2) + O(n). What is the general kth term in this case? And what value of k should be plugged in to get the answer?

- because $T(n)=3T(\frac{n}{2})+O(n)$
- $T(n)≤3T(n/2)+cn $ $ ≤3[3T(\frac{n}{4})+\frac{cn}{2}]+cn=9T(\frac{n}{4})+\frac{3cn}{2}+cn$ $ ≤9[3T(\frac{n}{8})+\frac{cn}{4}]+\frac{5cn}{2}=27T(\frac{n}{8})+\frac{3cn}{2}+cn+\frac{9cn}{4}$ $ ≤27[3T(\frac{n}{16})+\frac{cn}{8}]+\frac{19cn}{4}=81T(\frac{n}{16})+\frac{3cn}{2}+cn+\frac{9cn}{4}+\frac{27cn}{8}$
- As we can see,The general Iterm is $T(n)≤3^kT(\frac{n}{2^k})+2cn((\frac{3}{2})^k-1)$
- k should be plugged is $k=log{_2}{n}$, ***Reason: n is power of b, and b is 2 in this question. the depth is $k=log{_b}{n}$***

(b) Now try the recurrence T (n) = T (n – 1) + O(1), a case which is not covered by the master theorem. Can you solve this too?

- Because $T(n)=T(n-1)+O(1)$
- $T(2)=T(1)+ O(1)$ $T(3)=T(2)+ O(1)=T(1)+ 2O(1)$ $T(4)=T(3)+ O(1)=T(1)+ 3O(1)$
- As we can see $T(n)=T(1)+ (n-1)O(1)$
- because $O(1)$ is a constant, we can let $O(1)$ be c, then $T(n)=T(1)+c(n-1)$. Apparently $T(n)=O(n)$

2.5.

Solve the following recurrence relations and give a Θ bound for each of them.

- a. $T(n) = 2T(n/3) + 1$
  - $d=0$, $a=2$, $b=3$
  - $log{_b}{a}=log{_3}{2}>d$
  - $T(n)=O(n^{log{_3}{2}})$
- b. $T(n) = 5T(n/4) + n$

- $d=1$ $a=5$ $b=4$
- $log{_b}{a}=log{_4}{5}>d$
- $T(n)=O(n^{log{_4}{5}})$

- c. $T(n) = 7T(n/7) + n$
  - $d=1$ $a=7$ $b=7$
  - $log{_b}{a}=1=d$
  - $T(n)=O(n{log{n}})$

- d.$T(n) = 9T(n/3) + n^2$
  - $d=2$ $a=9$ $b=3$
  - $log{_b}{a}=log{_3}{9}=2=d$
  - $T(n)=O(n^2{log{n}})$

- e.$T(n) = 8T(n/2) + n^3$
  - $d=3$ $a=8$ $b=2$
  - $log{_b}{a}=log{_2}{8}=3=d$
  - $T(n)=O(n^3{log{}{n}})$

- f.$T(n)=49T(n/25)+n^\frac{3}{2}log{n}$
  - $d=3$, $a=49$, $b=25$, $f(n)=n^\frac{3}{2}log{n}$
  - let's take $g(n)=n^\frac{3}{2}≤f(n)=n^\frac{3}{2}log{n}$
  - In this case $d=3/2$
  - $log{*b}{a}=log*{{25}}{49}< d=3/2 $
  - so if $g(n)=n^\frac{3}{2}$,Then $T(n)=O(n^\frac{3}{2})$
  - $n^\frac{3}{2}log{n}$ has larger growth rate than $ n^\frac{3}{2}$
  - So $T(n)=n^\frac{3}{2}log{n}$

- g.$T(n)=T(n−1)+2$
  - $T(2)=T(1)+2$
  - $T(3)=T(2)+2=T(1)+4$
  - $T(4)=T(3)+2=T(1)+6$
  - $T(n)=T(1)+2(n-1)$
  - As we can see T(1) is a constant, f(n)=2(n-1) which has a larger growth rate
  - Then $T(n)=O(n)$

- h.$T(n)=T(n−1)+n^c,where c≥1isaconstant$

- $T(2)=T(1)+2^c$

- $T(3)=T(2)+3^c=T(1)+2^c+3^c$

- $T(4)=T(3)+4^c=T(1)+2^c+3^c+4^c$

- $T(k)=T(1)+2^c+3^c+4^c+...+k^c$

- Apparently,every time when n increase by 1,there will be a $(n+1)^k$, so k^c determine the growth rate $T(n)=O(n^c)$

2.22

You are given two sorted lists of size m and n. Give an O(log m + log n) time algorithm for computing the kth smallest element in the union of the two lists.

```
#again, we can input our R code here.
```