Jiacheng Yu 250799525

2.1

(a)Show that insertion sort can sort the n/k sublists, each of length k, in Θ(nk) worth-case time:

Insertion sort takes: ak^2=bk+c for some constants a,b,c

For n/k sublists $\frac{n}{k}(ak^2 + bk + c) = ank + b + \frac{cn}{k} = \Theta(nk)$ (shown)

(b) To merge n sublists of length k:

$$T(n) = \begin{cases} 0 & \text{if } n = 1 \\ 2T(n/2) + nk & \text{if } n = 2^a \text{ for } a > 0 \end{cases}$$

Merging n sublists is to divide them into n/2 lists in two groups merge each of it recursively that takes T(n/2), and then takes $\frac{n}{2}$ k steps to combine them together. Since there are tow arrays, it takes 2T(n/2) + nk in total.

Prove:

BASE: If there is only one list: T(1) 1k lg1 = 0

INDUCTION: Assume that T(n) = nk lg n is true, so for T(2n):

$$\begin{aligned} T(2n) = 2T(n) + 2nk = 2(T(n) + nk) &= 2(nk \lg n + nk) \\ &= 2kn(\lg n + 1) \\ &= 2nk(\lg n + \lg 2) \\ &= 2nk \lg(2n) \end{aligned}$$

(proven)

When $\frac{n}{k}$ lists, substitute into the equation, we have:

$$\begin{aligned} T(\frac{n}{k}) &= \frac{n}{k} k \lg(\frac{n}{k}) \\ &= n \lg(\frac{n}{k}) \end{aligned}$$

Hence, in worst case time, the complexity of merge the sublists is Θ(n lgn)

(c) When k is the largest, the running time is the same:

$$\begin{aligned} \Theta(nk + n\lg(n/k)) &= \Theta(n \lg n) \\ nk + n\lg(n/k) &= \lg n \\ k + \lg(n/k) &= \lg n \\ k &= \lg n \end{aligned}$$

If k > lgn, then the complexity will be greater than Θ(n lgn) ,

Hence k = lgn is the largest for the modified algorithm has the same running time as the sandard merge sort.

(d) see question5 part c

3.2

	A	B	O	o	Ω	ω	Θ
a	$\lg^k n$	n^{ε}	yes	yes	no	no	no
b	n^k	c^n	yes	yes	no	no	no
c	$\sqrt{n}$	$n^{\sin n}$	no	no	no	no	no
d	2^n	$2^{n/2}$	no	no	yes	yes	no
e	$n^{\lg c}$	$c^{\lg n}$	yes	no	yes	no	yes
f	lg(n!)	$Lg(n^n)$	yes	no	yes	no	yes

4.2

Binary Search

(a) By master method, when Time = Θ (1)

T(n) = T(n/2) + c

a = 1, b = 2, and $n^{\log_b{}^a} = n^{\log_2{}^1} = n^0 = 1$,case 2 applied

Hence, T(n) = Θ (lg n)

(b) When Time = Θ (N)

T(n) = T(n/2) + cN

The complexity is N times the case in (a), so T(n) = cN lg n = Θ (nlg n)

(c) T(n) = T(n/2) + cn, by master method

a = 1, b = 2, f(n) = cn, $n^{\log_b{}^a} = n^{\log_2{}^1} = n^0 = 1$, case 1 applied

Hence, T(n) = Θ (n)

Merge sort

(a) By master method,

T(n) = 2T(n/2) + cn

a = 2, b =2, and $n^{\log_b{}^a} = n^{\log_2 2} = n^1 = n$,case 2 applied

Since f(n) = Θ (n), T(n) = Θ (n lg n)

(b) When Time = Θ (N), T(n) = 2T(n/2) + cn + 2N

The complexity is $\Theta(n^2)$

(c) $T(n) = 2T(n/2) + cn + 2n/2$, by master method

a = 2, b =2, and $n^{\log_b{}^a} = n^{\log_2 2} = n^1 = n$,case 2 applied
Since $f(n) = \Theta(n)$, $T(n) = \Theta(n \lg n)$

Question 5:

(d) We do not want to run a) on input of size 200,000,000 because it will take too much time to complete compare to b).