Solution to Homework 1

2.3-3

Induction base case: for k=1, which means n=2. $n\log n$ gives 2log2=2, which means the base case holds.
Hyposthesis: suppose the solution holds for k,
Induction Step: For k+1, which means $n=2^{k+1}$, we have:
$T(2^{k+1})=2T(2^{k+1}/2)+2^{k+1}=2T(2^k)+2^{k+1}$.
By the Induction Hypothesis, we have: $T(2^k)=2^k \lg 2^k$.
Thus $T(2^{k+1})=2(2^k \lg 2^k)+2^{k+1}=2^{k+1}(\log 2^k+1)= 2^{k+1}(k+1)= 2^{k+1}\log(2^{k+1})=n\log n$.
So the result holds for all $k > 1$.

2.3-4 (for cs363)

T(n) = O(1), if n=1;
T(n-1)+O(n), if n>1.

So $T(n)=O(n^2)$+

2.3-7 (for cs463)

First, we can sort the array using merge-sort. This takes time of $\Theta(n \log n)$.
Then, implement the next procedure:

```
PROCEDURE SearchNumberPair(A[1..n], x)
    found←false
    for j←1 to n-1
        do
           key←A[j]
           result=BinarySearch(A[1..n], x-key)
           If result= true return(true)

        return(false)
```

BinarySearch has a running time of O(log$n$), thus this procedure costs n* O(log$n$)= O($n$log$n$).
So the total running time is: $\Theta(n \log n)$+O($n$log$n$)= $\Theta(n \log n)$.

2-4 Inversions

a. (1,5), (2,5), (3,5), (4,5), (3,4). (or: (2,1), (3,1), (6,1), (8,1), (8,6)
b. The array: [n, n-1, n-2, … , 2, 1].
   Number of inversions = (n-1)+(n-2)+…+1=n(n-1)/2
c. T(n)= $\Theta(n)+\Theta(\#\, inversion)$

d. Modify the merge-sort procedure, and calculate inversions during the merge-sort process and merge process:
   (1) during Merge-sort(A, p, r)
   #inversions in Merge-sort(A,p,r)=#inversions in Merge-sort(A,p,q) +
   #inversions in Merge-sort(A,q+1,r)+
   #inversions in Merge(A,p,q,r)
   (2) during Merge(A,p,q,r),

```
j←1
i←1
#inversions ←0
for k←p to r
    do if     L[i]<=R[j]
        then A[k] ←L[j]
                i←I+1;
        else  L[I] ←R[j]
              #inversions←#inversions+r-I+1
              j←j+1;
```

Obviously, the additional running time in calculating inversions is no more than $\Theta(n \lg n)$. Thus the total running time is:

T(n)= $\Theta(n \lg n) + \Theta(n \lg n) = \Theta(n \lg n)$

4-1 Recurrence Examples

a. Master Therorem case 3. T(n)= $\Theta(n^3)$
b. Master Therorem case 3. T(n)= $\Theta(n)$
c. Master Therorem case 2. T(n)= $\Theta(n^2 lgn)$
d. Master Therorem case 3. T(n)= $\Theta(n^2)$
e. Master Therorem case 1. T(n)= $\Theta(n^{lg7})$
f. Master Therorem case 2. T(n)= $\Theta(n^{1/2} lgn)$
g. Using iteration.
   $T(n)=n+T(n-1) = n+n-1+\ldots+2+T(1) =n(n+1)/2 + \Theta(1)= \Theta(n^2)$
h. Changing variables
   Let m=lgn, thus we have
   $T(2^m)=T(2^{m/2}) + 1$
   Rename $S(m)=T(2^m)$, then
   $S(m)=S(m/2) + 1$
   Using Master Theorem case 2, we get S(m)= $\Theta(lgm)$
   $T(n)=T(2^m)=S(m)= \Theta(lgm)= \Theta(lglgn)$