2.3-5

~~for j=1:n~~

~~Insertion Sort (Arr[1:n-1])~~

```
void Insertion Sort ( Arr, n ):
{  if n≤1 , return ;
   Insertion Sort (Arr, n-1) ;
   int j = n-2 , int last = arr[n-1]
   while ( j ≥ 0 && arr[j] > last )
   {  arr[j+1] = arr[j] ;
      j--;
   }
   arr[j+1] = last
}
```

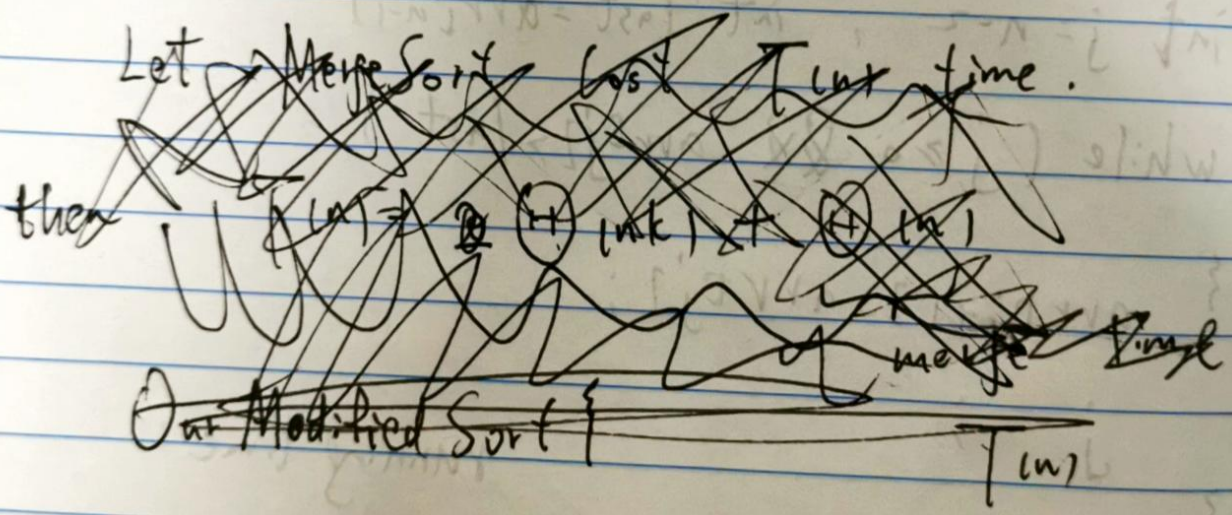
running time:

$$T(n) = \begin{cases} T(n-1) + O(n) & n \neq 1 \\ 1 & n = 1 \end{cases}$$

2-1

a. Using insertion sort to sort one length $k$ sublist will cost $\Theta(k^2)$ time, then, there are $n/k$ sublists. so will cost $\Theta(k^2 \cdot \frac{n}{k}) = \Theta(nk)$ in total

b. Let MergeSort cost $T(n)$ time.




$$T\left(\frac{n}{k}\right) = 2T\left(\frac{n}{2k}\right) + O(n)$$

$$\Rightarrow T(n) = \Theta(n \lg(\tfrac{n}{k}))$$

e.

c. Standard merge sort : $\Theta(n \lg n)$

when $k=1$. It has the same runing time as standard merge sort

d. let $f(k) = nk + n\lg(\frac{n}{k})$

$$f'(k) = n + n \cdot \frac{k}{n} \cdot (-\frac{n}{k^2}) = n - \frac{n}{k}$$

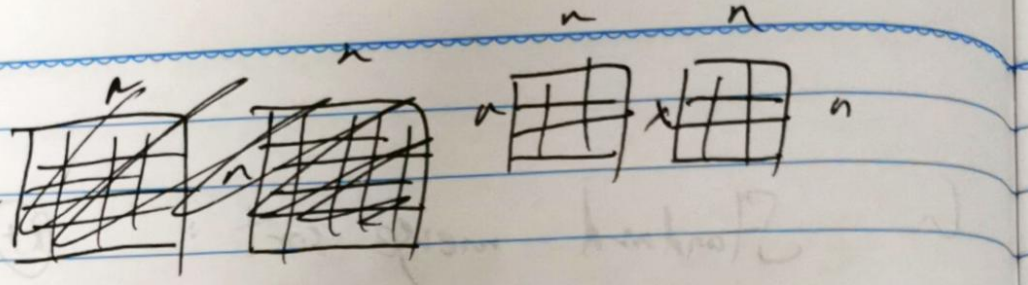
so $f$ $\begin{cases} \searrow & k \in (0,1) \\ \nearrow & k \in [1,n) \end{cases}$

so $f$ has minimum with $k=1$

$f_{min} = 1$ . However, I will choose $k$ with experiment's result.

P.m 27/八

4.2-3




$$T(n) = kT(\frac{n}{3}) + O(n^2) \quad \cdots [1]$$

$$\Rightarrow T(n) = k(kT(\frac{n}{9}) + O(\frac{n^2}{9})) + O(n^2)$$

$$\cdots \Rightarrow T(n) = k^m T(\frac{n}{3^m}) + (1 + \frac{k}{9} + \frac{k^{m-1}}{9^{m-1}}) O(n^2)$$

Let $m = \log_3 n + 1$

$$\Rightarrow T(n) = k^{\log_3 n + 1} \cdot T(3) + \frac{(\frac{k}{9})^{\log_3 n + 1} - 1}{\frac{k}{9} - 1} O(n^2)$$

$$= k^{\log_3 n + 2} + \frac{(\frac{k}{9})^{\log_3 n + 1} - 1}{\frac{k}{9} - 1} O(n^2)$$

$$= O(n^{\lfloor 97 \rfloor})$$

$$\Rightarrow \log_3 k < \lfloor 9 \rfloor$$

$$\Rightarrow K < 3^{\lfloor 9 \rfloor}$$