1. $T(n) = T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 1$

guess: $T(n) = O(n)$. Assume that $T(k) \le ck - d$ for $k \le \lceil \frac{n}{2} \rceil$ $(d \ge 0)$

then, $T(n) \le (c\lfloor \frac{n}{2} \rfloor - d) + (c\lceil \frac{n}{2} \rceil - d) + 1$

$= cn - 2d + 1 \le cn - d.$ $(d \ge 1)$.

$\therefore$ by induction, $T(n) = O(n)$.

2.

(a) $T(n) = 3T(\frac{n}{4}) + n^{\frac{1}{2}}$

by master method: since $n^{\frac{1}{2}} = O(n^{\log_4 3 - \varepsilon})$

$\Rightarrow T(n) = \Theta(n^{\log_4 3})$

(b) $T(n) = 2T(\frac{n}{2}) + n^3$

by master method: since $n^3 = \Omega(n^{\log_2 2 + \varepsilon})$, $T(n) = \Theta(n^3)$

(c) $T(n) = 8T(\frac{n}{4}) + n$

by master method: since $n = O(n^{\log_4 8 - \varepsilon})$, $T(n) = \Theta(n^{\log_4 8}) = \Theta(n^{\frac{3}{2}})$

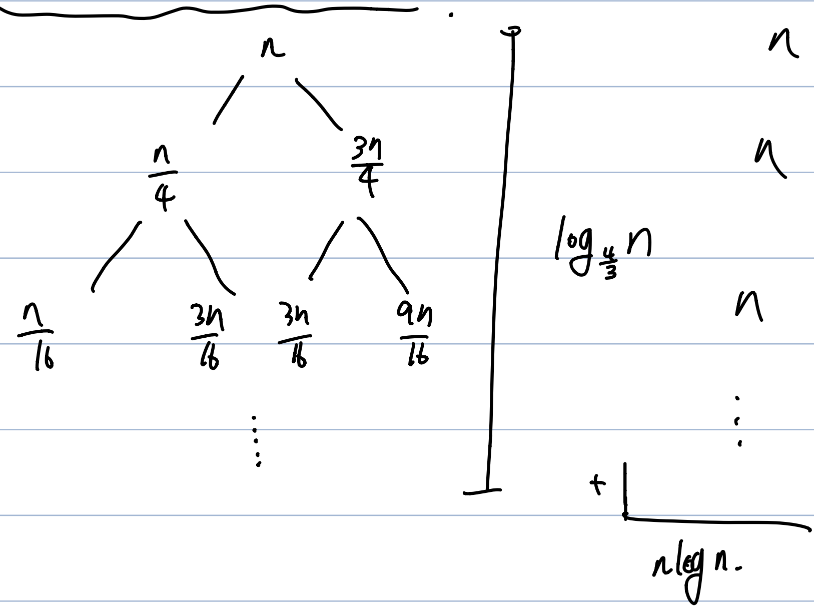
(d) $T(n) = T(n-3) + n^2$

$= T(n-6) + (n-3)^2 + n^2 = n^2 + (n-3)^2 + (n-6)^2 + \cdots \le \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$ $\therefore T(n) = O(n^3)$

그리고 $n^2 = \frac{3n^2}{3} \ge \frac{n^2 + (n-1)^2 + (n-2)^2}{3}$. 이를 반복하면 $T(n) \ge \frac{1}{3}\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{18}$ $\therefore T(n) = \Omega(n^3)$

$\therefore T(n) = \Theta(n^3)$

3. $T(n) = T(\frac{n}{4}) + T(\frac{3n}{4}) + O(n)$



$\therefore \quad T(n) = O(n\log n).$

4.

```
MergeSort (A, p, s)
    if  p<s
        q= ((s-p+1)/3)+p
        r= ((s-p+1)/3)+q
        MergeSort (A, p,q)
        MergeSort (A, q+1, r)
        MergeSort (A, r+1, s)
        Merge (A, p, q, r, s)
```

```
Merge(A, p, q, r, s)
    n1 = q - p + 1
    n2 = r - q
    n3 = s - r
    let L[1, ..., n1+1], M[1, ..., n2+1] and R[1, ..., n3+1] be new arrays.
    for i = 1 to n1
        L[i] = A[p + i - 1]
    for j = 1 to n2
        M[j] = A[q + j]
    for k = 1 to n3
        R[k] = A[r + k]
    L[n1+1] = ∞
    M[n2+1] = ∞
    R[n3+1] = ∞
    i = 1
    j = 1
    k = 1
    for l = p to s
        if L[i] ≤ M[j]
            if L[i] ≤ R[k]
                A[l] = L[i]
                i = i + 1
```

else

A[l] = R[k]

k = k+1

else

if M[j] ≤ R[k]

A[l] = M[j]

j = j+1

ele

A[l] = R[k]

k = k+1

장점 : 병렬 처리가 가능한 경우 (여러개의 비교를 동시에) 기존의 merge sort 보다 빠르다.

단점 : 병렬처리가 불가능 한 경우 time complexity가 $n\log_3 n$으로 더작지만, 실질적인 if문 (비교) 개수 가 더 많아서 오래걸린다.

5.

1st test : MN개의 data로 sort ⇒ best, avg : $O(MN \log MN)$

worst : $O((MN)^2)$

2nd test : $O(N^2) * M$ ⇒ $O(MN^2)$ best : $O(MN)$

∴ best : $O(MN \log MN)$

avg : $O(MN \log MN + MN^2)$

worst : $O((MN)^2)$