سوال ۱ : فرض می‌کنیم $n=2^k$ و طبق فرمول ارائه داریم:

$$T(n) = 2T(n/2) + \frac{n}{\log(n)} = \rightarrow \left(\sum_{i=1}^{n} (1/i) \simeq \log n\right.$$

$$2\left(2T(n/4) + \frac{n/2}{\log(n/2)}\right) + \frac{n}{\log(n)}$$

$$= 4t(n/4) + \frac{n}{\log(n/2)} + \frac{n}{\log(n)}$$

$$= 4\left(2t(n/8)\right) + \frac{n/4}{\log(n/4)} + \frac{n}{\log(n/2)} + \frac{n}{\log(n)}$$

$$= n \times t(1) + \frac{n}{\log(2)} + \frac{n}{\log(4)} + \cdots + \frac{n}{\log(n)}$$

$$= n\left(1 + \sum_{i=1}^{\log n} \left(\frac{1}{\log(2^i)}\right)\right.$$

$$= n\left(1 + \sum_{i=1}^{\log n} (1/i)\right) \simeq n(1 + \log(\log(n)))$$

$$= n + n\log(\log(n)) \simeq n \times \log(\log(n))$$

$$\Rightarrow O(n \log(\log n))$$

Subject:
Date

$$T(n) = \sqrt{n}\, T(\sqrt{n}) + O(\log n)$$

$$n = 2^m$$

$$T(2^m) = 2^{m/2} T(2^{m/2}) + O(m)$$

$$T(m) = \underbrace{m/2 + T(m/2)}_{f(n)} + \underbrace{O(m)}_{g(n)}$$

$$f(n) \Rightarrow m^{\log_2 2^{m/2}} = m^{\log_2 m - \log_2 2} = m^{\log_2 m - 1}$$

$$g(n) \Rightarrow m < m^{\log_2 m - 1}$$

طبق قضیه اصلی داریم:

$$\Theta(m^{\log m - 1})$$

حال با جایگذاری $m = \log(n)$

$$\Theta(\log n^{\log_2 \log n - 1})$$

Subject:
Date

سوال 2 :

برای پیاده سازی یک ساختار داده ای نیاز داریم که بتوانیم به صورت بهینه :

دو جلوترین عناصر لیست دسترسی داشته باشیم .

از min_heap استفاده می کنیم

و پیچیدگی زمانی $O(\log n)$ است .

Subject:
Date

سوال $\frac{3}{5}$:

روش حل استفاده از Disjoint-set ها می‌باشد. برای حل مراحل زیر را طی می‌کنیم:

1- یک آرایه‌ای به اندازه n تعریف می‌کنیم به تعداد تمای MODE ها است

2- برای هر index شماره i در آرایه مقدار رأس پدر i می‌دهیم.

3- هر نود به عنوان پدر خودش معرفی می‌شود.

```
int merge(int *parent, int x){
    if (Parent[x] == x)
        return x;
    return merge(parent, parent[x])
}

int ConnectedComponents(int n, vector<vector<int>> & edges)
{
  int parent[n];
  for(int i=0; i<n; i++){
      parent[i] = i;
```

PAPCO

ادامه

Subject:
Date

```
for (auto x: edges) {
    parent[merge(parent, x[0])] = merge(parent, x[1]);
}
int ans = 0
for (int i=0; i<n; i++) {
    cns += (parent[i] == i); {
for(int i = 0; i<n; i++){
    Parent[i] = merge(parent, Parent[i])
}
map<int, list<int>> m;
for(int i=0; i<n; i++){
    m[parent[i]].push_back(i);
}
```

mbl

Subject:
Date

ادامه

```
for( auto it = m.begin () ; it != m.end() ; it++)
{
    list<int> l = it -> second
    for(auto x : L) {
        Print ( x) ;
    }
}
return ans;
}
```

← این کد تعداد مولفه‌هایی که وصل هستند و به‌ترتیب node ها را به‌عنوان خروجی به ما می‌دهد.

← مرتبه زمانی این کد O(n) می‌باشد