Assignment -4 -Quiz

Tabeed L1F17BSCS0516

Problem 1:

B)

$T(n)=T(n/2)+T(n/2)+C$

$T(n/2)=T(n/4)+T(n/4)+C$

$T(n/4)=T(n/8)+T(n/8)+C$

Total N Entries so n*logn=nlogn

Problem 2:

Solution :

A)

```
    selectionSort(vector<int>v, int n)
    if n is greater than 1  base case
    max = FindMaxIndex(v, n) find  Max
    swap(v, max, n - 1)  swap Max with last index
    selectionSort(v, n - 1) recursively for n-1
```

B)

$T(N)= T(n-1) + n + 1$

C)

T(n) = T(n-1) + n

T(n-1) = T(n-2) + n

T(n-2) = T(n-3) + n

-> n + (n-1) + (n-2) +(n-3) + ... + 2 + 1 -> n(n+1)/2

by back substitution, we get n + n + n + n + ...

n(n+1)/2 → n^2

complexity is O(n^2)

Problem 3:

(a) T(n) = 4T(n/7) + 1

Series of Cost :

1+4^1(1)+ 4^2(1)+ 4^3(1)......4^logN(1)

1+4^1(1)+ 4^2(1)+ 4^3(1)...... Nlog^4

Increasing series

Cost = O(N)

(b) T(n) = 3T(2n/3) + 2n.

increasing series

Cost =O(3log3^2n)

(c) T(n) = 3T(n/3) + 2n.

Cost = O(nlogn)

(d) T(n) = T(n) + 1, and T(2) = 1.

E)

i) $T(n) = 4T(n/2) + 5n^2$

Cost = O(n^2logn)

ii) $T(n) = 4T(n/2) + 10n^2$

Cost = O(n^2logn)

Problem 4:

Solution :

Assuming Array {2,0,0,1,2,1,0,2,1,0,1,1,0}

Expected output {0,0,0,0,0,1,1,1,1,1,2,2,2}

Zero Means= RED

One Means= White

Two Means= BLUE

we have to Sorted The data in Such a way that RED Comes Before White and BLUE Should Come after

Enum={Red,Blue,White};

Red_White_Blue_Sort(vector<int>keys, int n)

low = 0, high = n-1 mid = 0;

```
while (mid <= high)
        If Keys[mid] == Red =>swap(key [low++], key [mid++]) break
        If Keys[mid] == White => mid ++  break
        If Keys[mid] == Blue =>swap(key [mid], key [high--]) break
```

one traversal of the Array will sort them

Swap in O(1)

So O(N) Time Complexity