Grand Quiz (F)

Reg. No. L1f17BSCS0516_____
Time: 40 mins.

Q1: (15 Marks) Solve the following recurrence relation ***showing all steps***. Also give the Asymptotic Complexity.
Do not use Master's method.

$$\mathbf{T(n)} = \begin{cases} c\,1; if\ n=1 \\ 3T\left(\frac{n}{3}\right)+cn^2; if\ n>1 \end{cases}$$

Ans:

T(N) ..... N^2

T(N/3) T(N/3) T(N/3) .....3(N^2/3^2) = N^2/3

It's a decreasing order series so by master theorem
Cost = Cost at First Level = O(N^2)

Grand Quiz (F)

Reg. No. L1f17BSCS0516_____
Time: 40 mins.

Q2: (25 Marks)
We have n elements in an array; there are only two distinct numbers in the array, i.e. all the elements are copies of two numbers. Array is sorted in non-increasing order. Propose an algorithm to find the **number with lesser occurrences** using **divide and conquer** technique. Discuss the solution then write it in the proper function form in Pseudocode with all the conditions and checks. Give asymptotic complexity of your algorithm.

Ans:

Sample input = {5,5, 4.3,3,2,2,2,1}

Count_Occurences(vector<int>v) → which has the above value

Maps<- first we will make a map of <int,int> to store the count of every element

map<int,int> count // this will count the each element **occurrences**
now we will loop through it
for i=0 to i<v→size // will run till vector.size

count[v[i]]++
// this single line will count all the accurence of a single number in the data
now we will simply print the data
for (auto x : count)
cout “element” << x . first << “ ” << “ accure “ << x . second << “\n”