***Name: Muhammad Sajid***

***Roll NO : 2***

***Class : MSc final (Morning)***

**Paper: 2018**

**Q ii (b)**

*RECURSION:*

***Recursion is the method for solving the problem, where in recursion the original problem is divided into smaler problem of the same instance of the original problem.***

***The exprasion for recursion***

***T(n) = 2T(n/2) + n, where 2T ,here 2 is the number of sub problem and n/2 , 2 is the division of the problem.***

***Solving the recurrence equition for Merge sort.and we apply the master theorem to solve and find the time comlexity of Merge sort.***

***Case 1: Ѳ(nlogba ) ,where f(n) = O(n\*log ba-ɛ)***

***Case 2: Ѳ(nlogba) ,where f(n) = Ѳ ( n\* log ba )***

***Case 3: Ѳ(f(n) ,where f(n) = Ω(nlogba+ɛ) and f(n/b)< c f(n) for large n***

***The recurrence Equation for Merge sort***

***T(n) = 2 T ( n/2 ) + n***

***Here a=2 , b=2 and f(n) = n, here the sub problem is 2 and it is divide by 2***

***So f(n) = nlogba ==> nlog 22***

***= f(n) = n1***

* ***f(n) = n Since case 2 are applied from the above***

***Since f(n) = Ѳ(nlogba)***

***Thus the soloution is T(n) = Ѳ(nlogba \* log n)***

***Where f(n) = Ѳ(nlogba)***

***T(n) = Ѳ(nlog22 \* log n)***

***T(n) = Ѳ (n1\* log n )***

***Simplized that T(n) = Ѳ(n \* log N)***

***Paper-2018***

***Q II***

***And (b):***

***Recurrence relation for the time complexity of the merge sort for worst case .***

* ***Using any Method to finde the time complexity***

***Sol:***

***We know that equation for Merge sort***

**T(n) = 2T ( n/2 ) + n**

**Here a=2 , b=2 and f(n)= n**

**We solve using 2nd form of Master theorem**

**F(n) nlogba 🡺 nlog22**

**F(n) n**

**F(n) = n So case 2 are applied ,**

**Since f(n) = Ѳ (nlogba)**

**Thus the solution is**

**T(n) = Ѳ ( nlogba log n )**

**Where f(n) = Ѳ ( nlog22 log n )**

**T(n) = Ѳ ( n log n )**

**T(n) = Ѳ ( n log n)**

**This is the time complexity of Merge Sort**

**Paper- 2016**

**Q II**

**Ans : (a)**

**The problem that are base upon the divide and conqure approuch in the which the problem are divide into subproblem of the original problem and it is solved.**

**The example are fibonacii , factorial and Merge sort etc problems.**

**Ans (b):Merge sort in term of Recurrence**

**We know the equation for merge sort**

**T(n) = 2T ( n/2 ) + cn -----> A**

**T( n/2 ) = 2T (n/2 \* 1/2) + cn/2**

**T(n/2) = 2T (n/4) +cn/2**

**Putting in the equation in A**

**T(n) = 2[ 2T ( n/4 ) +cn/2 ] + cn**

**T(n) = 4T(n/4) + 2\* cn/2 + cn**

**T(n) = 4T(n/4) +2cn---------> B**

**T(n/4) = 2T(n/2 \* 1/4) + cn/4**

**T(n/4) = 2T ( n/8 ) + cn/4**

**Put in Equation B**

**T(n) = 4[2T(n/8)+cn/4]+ 2cn**

**T(n) = 8T(n/8) + 4cn/4 +2cn**

**T(n) = 8T(n/8) + cn +2cn**

**T(n) = 8T(n/8)+3cn**

**…….**

**And so on**

**We have the Genral Equation**

**T(n) = 2k . T ( n/2k ) + k cn**

**Paper-2016**

**Q II**

**Ans (b):**

**Master theorem**

**T(n) = aT (n/b ) + f(n) where a>= 1 , b>1 and f(n) > 0**

**We compare f(n) with nlogba**

**f(n) is asymptotically smaller or large than nlogba  by a polynomial factor nɛ**

**or**

**f(n) is asymptotically equal with nlogba**

**there are three cases in Master theorem**

**case 1:**

**if f(n) = o (nlogba-ɛ ) , then T(n) = o(nlogba )**

**case 2:**

**if f(n) = o (nlogba-ɛ ) , then T(n) = o(nlogba log n)**

**case 3:**

**if f(n) = Ω (nlogba+ɛ ) , for some ɛ > 0 , and if af(n/b) <= cf(n)**

**for some c<1 and all sufficiently large n ,**

**then T(n) = O (f(n))**

**Example:**

**We have**

**T(n) = 2T (n/2) + n2**

**Here a=2 , b=2 and f(n) = n2**

**So compare**

**f(n) = nlogba**

**f(n) = nlog22**

**f(n) = n**

**so f(n) > n2**

**since case 3 are applied in this case**

**f(n) = Ω ( nlog22+ɛ )**

**af(n/b) <= cf(n)**

**2 n2/4 <= cn2 🡺 ½ <= c<1**

**T(n) = O (f(n) ) = O(n2)**