Meet Raut
S21 84

TSEC

Assignment 1

Q.1)

a)

Ans. $T(n) = 2T(n/2) + n^3$

comparing with $T(n) = aT(n/b) + f(n)$

$a = 2$, $b = 2$, $f(n) = n^3$

~~nlog~~ $n^{\log_b a} = n^{\log_2 2} = n < f(n)\ (n^3)$

According to master theorem.

case 3:

If $f(n) = \Omega(n^{\log_b a + \epsilon})$ $(\epsilon > 0)$ & if $af(n/b) \le kf(n)$

Then $T(n) = \Theta(f(n))$

$af(n/b) = 2f(n/2)^3 = n^3/4 \le kn^3$

This is satisfied for $k = 1/2$

so by master theorem $\Theta(n^3) = T(n)$

b)

Ans $T(n) = T(9n/10) + n$

comparing with $T(n) = aT(n/b) + f(n)$

$a = 1$, $b = 10/9$, $f(n) = n$

$n^{\log_b a} = n^{\log_{10/9} 1} = 1 < f(n)$ (i.e n')

If falls in case 3 of master theorem

check if $af(n/b) \le kf(n)$, where $k < 1$

$1 \times (9/10) \le k.$ ~~n~~ $\Rightarrow k \geq 9/10$

It satisfies for $k = 19/20$

So by master's theorem $T(n) = \Theta(n)$

c) $T(n) = 16T(n/4) + n^2$

Ans by Comparing,

$a=16$, $b=4$, $f(n) = n^2$

$n^{\log_b a} = n^{\log_4 16} = n^2 = f(n)\ (n^2)$

According to second case of master theorem

$T(n) = \theta(n^2 \log n)$

d) $T(n) = 4T(n/2) + n^3 + 4n + 5$

Ans. $a=4$, $b=2$, $f(n) = n^3+4n+5$

$n^{\log_b a} = n^{\log_2 4} = n^2 < f(n)$ (i.e. n^3+4n+5)

So by case 3 of master theorem

If $af(n/b) \le kf(n)$, $k<1$

$4((n/2)^3 + 4(n/2) + 5)$

$\Rightarrow 4(n^3/8 + 2n + 5)$

$\Rightarrow n^3/2 + n/2 + 5/4 \le k(n^3 + 4n + 5)$

This holds true for $k > 1/8$

So by masters theorem, $T(n) = \theta(n^3 + 4n + 5)$

Incorrect

correct it

e)

Ans. $a=2$, $b=2$, $f(n) = n+1$

$n^{\log_b a} = n^{\log_2 2} = n < f(n)\ (n+1)$

This falls in case 3

If $af(n/b) \le k.f(n)$, $k<1$; $2(n/2+1) \le k(n+1) \Rightarrow n+2 \le k(n+1)$

This holds for $k = 1/2$

So, by master theorem $T(n) = \theta(n+1)$

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Q. 2)

a)

Ans. Using master's th^m

$a = 5, \quad b = 2, \quad k = 0, \quad p = 0$

$n^{\log_b a} = n^{\log_2 5} > f(n)$

This fall in case 1 of master's th^m

$f(n) = \cancel{O(n \log} \; O(n^{\log_2 5 - \epsilon}) \quad (\epsilon > 0)$

$\therefore T(n) = \theta(n^{\log_2 5})$

b)

Ans. $a = 1, \quad b = 4, \quad f(n) = n$

$n^{\log_b a} = n^{\log_4 1} = n^0 = 1 < n$ (i.e. $f(n)$)

we fall in case 3

$f(n) = O(n^{\log_b a + \epsilon}) \quad (\epsilon > 0)$

if $a f(n/b) < k . f(n) \quad k < 1$

$1 . (n/4) < k(n)$

This holds true for $k = 1/2$

$\therefore T(n) = \theta(f(n)) = \theta(n)$

Q. 3)

Ans. Assume $T(n) = O(n^k)$ for constant k

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

$T(n) = cn^k + n = c(n/2)^k + n = \frac{cn^k}{2^k} + n$

we want to prove, $T(n) = O(n^k)$, let's try to bound

$T(n) \leq \frac{cn^k}{2^k} + n$

let $c/2^k$ be constant c

$T(n) \leq cn^k + n \leq c'n^k$ (say)

$k=1 \implies T(n) \leq Cn + n \leq C'n$

It holds true for $n \geq 1$ if $c' \geq 2c$

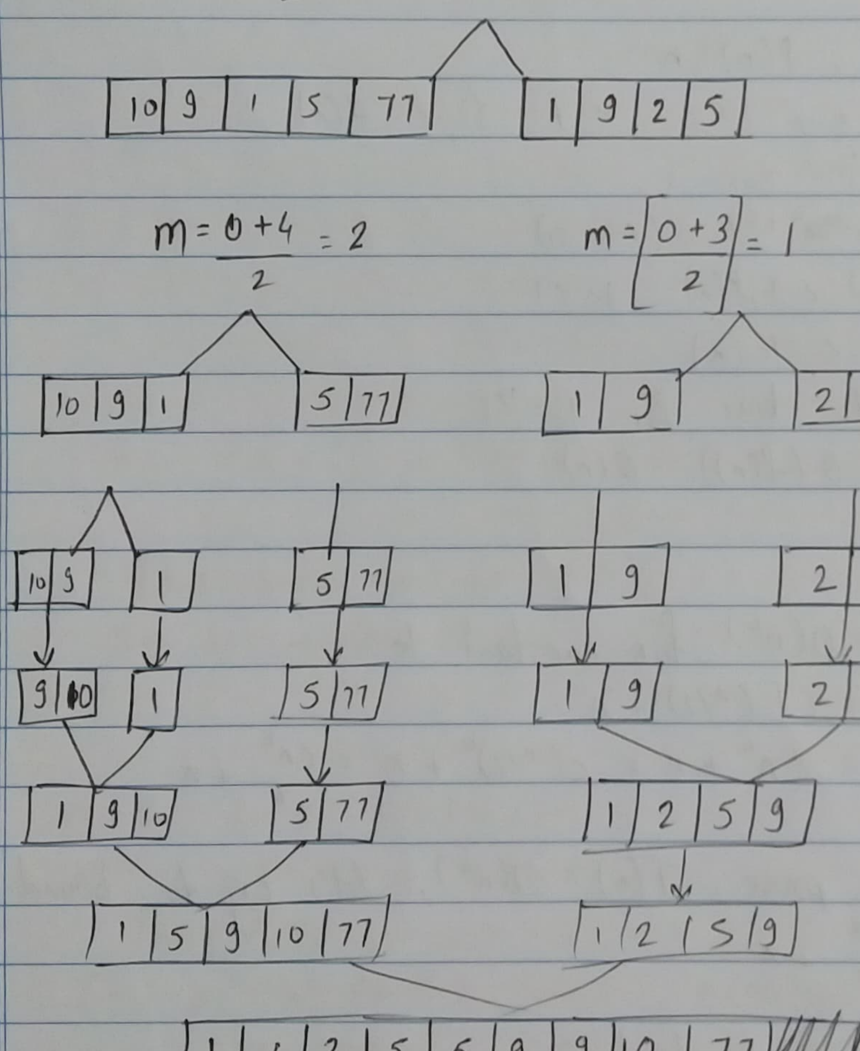
Hence, by induction,

$T(n) = O(n)$

Q.4)
Ans.

10	9	1	5	77	1	9	2	5
0	1	2	3	4	5	6	7	8

$$m = \frac{l+n}{2} = \frac{0+8}{2} = 4$$



Time Complexity : $2^k T(n/2^k) + k.n$

let $2^k = n \implies k = \log_2 n$

$n T(n/n) + n\log_2 n \implies O(n\log_2 n)$

Q. 5)

Ans. let first element be the pivot

0	1	2	3	4	5	6	
E	X	A	M	P	L	E	∞
i, pivot						j	

Well increment i till i < pivot and
decrement j till j > pivot and swap i & j.

E	X	A	M	P	L	E	∞
pivot	i				j		

E	X	A	M	P	L	E	∞
pivot	i	j					

E	A	L	M	P	X	E	∞
pivot	j	i					

Since, index[j] < index [i] , pivot = j

A	E	L	M	P	X	E	∞
pivot	i					j	

This won't sort, since A is in sorted position

A	E	L	M	P	X	E	∞
	pivot					j	

This won't sort, as well since E is sorted too

A	E	L	M	P	X	E	∞
		~~pivot~~ pivot	i			j	

A	E	E	L	P	X	M	∞
			pivot	i		j	

This won't sort

A	E	E	L	P	X	M
				pivot	i	j

A	E	E	L	P	M	X
				pivot	j	i

A	E	E	L	M	P	X

Best case : Occurs when pivot divides list into two equal sublist of equal sizes. $O(n \log n)$

Avg. case : On avg. pivot nearly divides list to nearly two equal parts $O(n \log n)$

Worst case : This occurs when pivot is always smallest or largest element in list $O(n^2)$

Q.6)

Ans.

```
I = 1;
While (I <= n) {
      X = X + I;
      I = I + 1;
}
```

Frequency Count :

Initialisation for I = 1

Increment for X and I combined = 2n

Loop Run = n

Time complexity :

Initialization = $O(1)$

While loop = $O(n)$

Increment for X & I combined = $O(n)$

Overall = $O(1) + O(n) + O(n) = O(2n+1) = O(n)$

Q.7)

Ans. i) Substitution method :

It involves guessing a solution and then solving by using mathematical induction.

ii) Recurrence tree method :

It involes representing recurrence of a tree where each node represents cost of single subproblem

iii) Master theorem :

It involes a straight forward way to solve recurrence of the form $T(n) = aT(n/b) + f(n)$

It's a powerful and widely used method to solve recurrence problem

iv) Iteration method:

This method involves expanding the recurrence into iterations until a pattern emerges.

v) Generating functions:

This are used to reprents sequence of number as power series. Recurrences can be translated into equations involving generation functions, which can be solved using algebric techniques

Q.8)

Ans.

P - Type	NP - Type
→ Set of decision problems that can be solved using turing machine	→ Set of decision problems for which a solution can be guessed efficiently.
→ Running Time of an algo is bounded by polynomial function size	→ If someone claims to be solution to NP problem, it can be verified quickly
→ Effectively soluable as running time grows polynomially with input.	→ A solution in NP can be guessed efficiently but maybe difficult for traditional
→ Ex. Sorting, searching, basic arithmetic operations	→ Ex. Travelling salesman problem, knapsack problem

B+