Assignment - 1

Name → Sanyam Jain

U. Roll no. → 2015273

C. Roll no. → 71

Section → ML

Signature → Sjain

Ans 1) Asymptotic notation are mathematical tools to repre the time complexity of algo for asymptotic analysis

The main idea of asymptotic analysis is to have a measure of the efficiency of algo that don't depend on machines.

Following are the asymptotic notations:-

(i) Θ - Notation → The Θ notation bounds a func from above and below, so it defines exact asymptotic behavio

(ii) Big O Notation → O define upper bound of a algo it bounds a function only from above

(iii) Ω Notation → Ω notation provide an lower bound

Ans 2) $\Theta(\log n)$

Ans 3) $T(n) = \begin{cases} 3T(n-1), & \text{if } n>0 \\ 1, & \text{otherwise} \end{cases}$

$T(n) = 3T(n-1)$

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

$= 3^2 T(n-2)$

$= 3^3 T(n-3)$

$=$

$=$

$=$

$3^n T(n-n)$

$= 3^n$

Ans 4) $T(n) = \begin{cases} 2T(n-1)-1, & \text{if } n>0 \\ 1, & \text{otherwise} \end{cases}$

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

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

$= 2^2(T(n-2)]-2-1$

$= 2^2(2T(n-3)-1)-2-1$

$= 2^3T(n-3)-2^2-2^1-2^0$

$= 2^n T(n-n) - 2^{n-1} - 2^{n-2} - 2^{n-3} - \cdots - 2^2 - 2^1 - 2^0$

$= 2^n - (2^n - 1)$

$= 2^n - 2^n + 1 = 1$

Ans 5) $S_i = S_{i-1} + i$

If k is total number of iterations taken the program then while loop terminates ⇒

$$1+2+3 - - - -k = [k(k+1)/2] > n$$

$$\therefore k = O(\sqrt{n})$$

Ans 6) $O(\sqrt{n})$

Ans 7) j is loop executing $\log n$ times

k " " " $\log n$ "

i " " " $n/2$ "

$$T.C = O(n \log^2 n)$$

Ans 8) $O(n^3)$

Ans 9) inner loop will execute $(n + \frac{n}{2} + \frac{n}{3} + \cdots \frac{n}{n})$

$$= n(1 + \frac{1}{2} + \frac{1}{3} + \cdots \frac{1}{n})$$

It is equal to $\Theta(\log n)$

Ans 10) $n^k \quad a^n$

$k >= 1 \quad a > 1$

Taking $k = a = 2$

$n^2 \quad 2^n$

we can say $n^2 = O(2^k)$

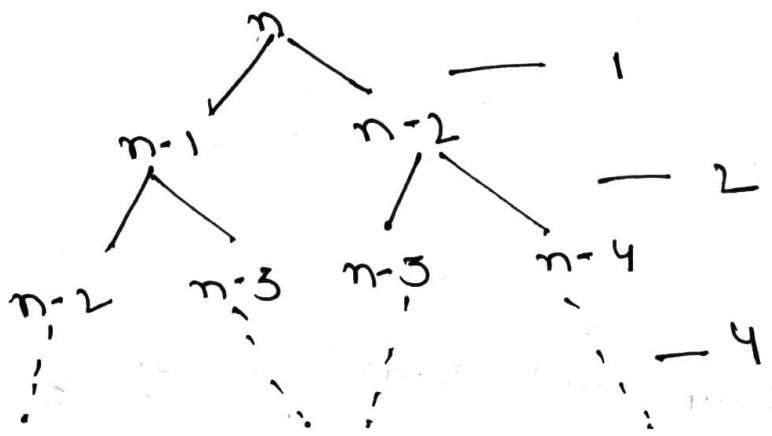
$n^k = O(a^n)$

Ans 11) $O(\sqrt{n})$

Ans 12) Recurrence relation

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

making Recurrence Tree



$$TC = 1 + 2 + 4 + \cdots + 2^n$$

$$a = 1, \quad r = 2$$

$$1 \frac{(2^{n+1} - 1)}{2 - 1} = 2^{n+1} - 1$$

$O(2^{n+1}) = O(2 * 2^n) = O(2^n)$

Space complexity = $O(n)$

This is because max stack frame is same equal to n only as frame is called like this

$f(n-1) + f(n-2)$

$f(n-2)$ is called when we get the return val of $f(n-1)$

∴ it is equal to $O(n)$

Ans 13) $n \log n$

```
for(i=1 ; i<n ; i++)
    for(j=1 ; j<n ; j = j+i)
        cout<<"#";
```

n^3

```
for(i=0 ; i<n ; i++)
    for(j=0 ; j<n ; j++)
        for(k=0 ; k<n ; k++)
            cout<<"#";
```

log log n

```
int fun(int n)
{
    if (n<=2)
        return 1;
    else
        return (fun(floor(sqrt(n)))+n);
}
```

Ans 14) $T(n) = T(n/4) + T(n/2) + n^2$

$T(n/2) >= T(n/4)$

$T(n) = 2T(n/2) + cn^2$

Apply master method

$a = 2, \quad b = 2$

$k = \log_b a = \log_2 2 = 1$

$n^k = n$

$f(n) = n^2$

as it $\Omega(n^2)$

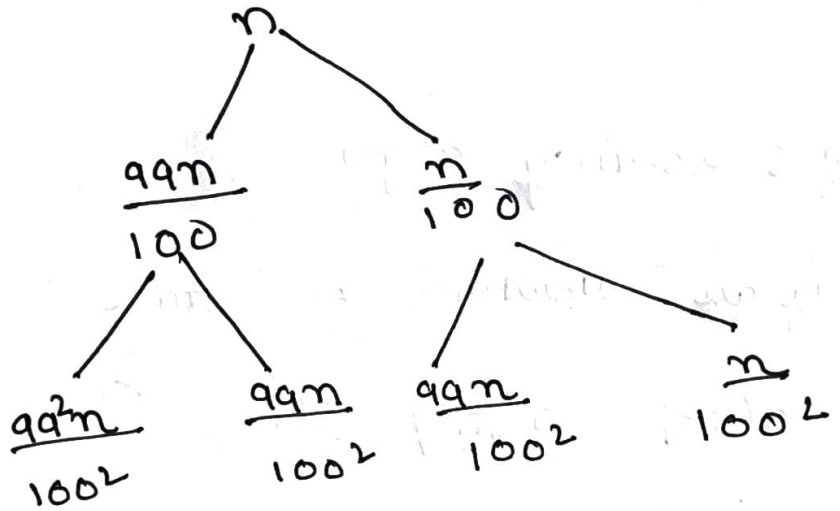
But as $T(n) <= \Omega(n^2)$

$T(n) = O(n^2)$

Ans16) if K is come greater than 1

then T.C = O(log log n)

Ans17) $T(n) = T\left(\frac{99n}{100}\right) + T\left(\frac{n}{100}\right)$



if we take longer brach i.e $\frac{99n}{100}$

TC = log 100/99 n log n

Ans18)

a) 100 log log n $\sqrt{n}$ n log n ! n log n n^2 2^n $2^{2n}/4^n$ n!

b) log log $\sqrt{\log n}$ log n 2 log n log 2n n 2n 4n log n!
n log n n^2 2(2^n) n!

c) 96 $\log_8 n$ 5n log n ! n $\log_6 n$ $\log_2 n$ $8n^2 + n^3$ 8^{2n}

Ans 19)

```
linear search ( array, key )
    for i in array
        if val == key
            return i
```

Ans 20/21/22)

Considering ~~only~~ 3 sorting algo

	Best case	Avg case	Worst case	SC	stable	Inplace	On
Bubble sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	✓	✓	✗
Selection sort	$O(n^2)$	$O(n^2$	$O(n^2)$	$O(1)$	✗	✓	✗
Insertion sort	$O(n)$	$O(n^2)$	$O(n^2)$	$O(1)$	✓	✓	✓

Ans 23)

Arr → array

n → size

x → val to be search

if upperbound < lowerbound

x does not exist

mid = (l.b + u.b) / 2

if Arr[mid] == X

x found at mid

if A[mid] > X

UB = mid - 1

if Arr[mid < X

LB = mid + 1

	TC	SC
Linear	O(n)	O(1)
Binary (Recursive)	O(log n)	O(log n)
Binary (Iterative)	O(log n)	O(1)

ANS 24) $T(n) = T(n/2) + C$