DAA ASSIGNMENT-1

Date .....

Ans 1:- Asymptotic notation are mathematical tools to represent the time complexity of algorithms for asymptotic analysis.

The main idea of asymptotic analysis is to have a measure of the efficiency of algorithm that don't depends on machine. Specific Constants & doesn't require algorithm to be implemented & time taken by the program to be compared.

Asymptotic notations that are mostly used:-

1) Θ-Notation:- The theta notation bounds a functions from above & below, so it defines exact asymtotic behaviour.

2) Big O Notation:- It defines an upper bound of an algorithm, it bound a function only from above.

3) Ω Notation:- Ω Notation provides an asymptotic lower bound.

- It takes linear time in best case & quandatic time in worst case
ex -> Conside Insertion sort.

we can say that Insertion sort have.
$O(n^2)$
$\Theta(n^2)$ for worst case
$O(n)$ for best case
$\Pi(n)$

Date .....

Ans 2) $O(\log n)$

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

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

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

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

$$\boxed{T(n) = 1}$$

Date .....

Ans 5) $Si = Si-1 + i$

if k is total number of iterations taken by the program then while loop terminates

$$1+2+3+\ldots+k = \left[\frac{k(k+1)}{2}\right] > n$$

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

Ans 6 :- $O(\sqrt{n})$ or $O(\log n)$

Ans 7) j is loop executing log n times
k " " " log n times
i " " " n/2 times

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

Ans 8) ~~$O(n^3)$~~ $O(n^3)$

Ans 9) ~~$O(n \log n)$~~ ~~$O(\log n)$~~ $O(n \log n)$

Ans 10) n^k a^n

$k >= 1$ $a > 1$

Taking $k = a = 2$

n^2 2^n

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

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

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

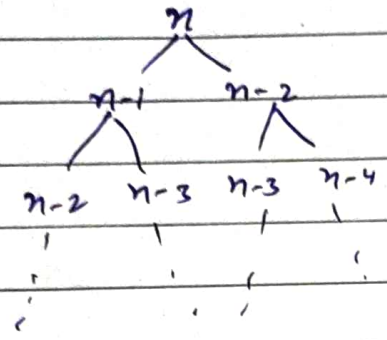
Spiral

Date .....

Ans-12) Recurrence Relation

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

making Recurrence Tree



$T.C = 1 + 2 + 4 + \dots + 2^n$

$a = 1 \quad r = 2$

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

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

$S.C = O(n)$

This is because maximum stack frame is equal to n only as function is called like this.

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

$f(n-2)$ is called when we get the return value from $f(n-1)$

It is equal to $O(n)$

Ans-16) $T.C = O(\log \log n)$

Date .....

Ans(3) $n \log n$

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

n^3:

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

$\log(\log n)$:

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

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

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

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

Apply master method.

$a=2 \quad b=2, \quad k = \log_b a = \log_2 2 = 1$

$n^k = n$

$f(n) = n^2$

It is $O(n^2)$

Both as $T(n) <= O(n^2)$

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

Spiral Ans 15: $O(n \log n)$

Date .....

Ans 18)

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

b) $1 \log \log_n \sqrt{\log_n} \; \log_n \; n \log_n \; \log 2^n \; n \; 2n \; 4n \; \log_n! \; n \log_n \; n^2 \; 2(2^n) \; n!$

c) $96 \log_{8^n} 5n \; \log_n! \; n \log_{6^n} \; n \log_{2^n} \; 8n^2 \; 7n^3 \; 8^{2n} n!$

Ans 19) Linear Search (array, key)

```
for i in array
    if value == key
        return i
```

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