Name → Mond Nasir
Section → F
Roll No → 14

Student Id → 20021595
University Roll → 2016855

DESIGN AND ANALYSIS OF ALGORITHM
TUTORIAL-01.

Ques→1 Asymptotic Notation :→ Asymptotic notations are the mathematical notations used to describe the complexity (runing time of an algorithm. Different types of asymptotic Notation.

(i) Big-O :→ Big O notation specifically describes worst case scenario. It represents the tight upper bound running time complexity of an algorithm.

$$f(n) \leq c \cdot g(n). \quad \forall \; n \geq n_0, \; c \geq 0$$

Ex :→ O(1), O(n), O(nlogn).

```
for (i=0; i<n; i++)
    sum = sum + i;
```

The time complexity of above example is O(n).

(ii) Omega (Ω) :- omega Notation specifically describe best case scenario. It represents the tight lower bound running time of an algorithm.

$$f(n) \geq c \cdot g(n) \quad \forall \; n \geq n_0, \; c \geq 0$$

Example :- $\Omega(1)$, $\Omega(\log n)$.

```
void search (int size) {
    int element, i;
    if (& size == 1) {
        Printf ("found");
        return;
    }.
    for (i=0; i< size; i++)
    {
        if (arr [i] = element) {
            Printf ("found");
            return;
        }
    }
    Printf ("Not found");
}.
```

The Best Complexity of above Example is :- $\Omega(1)$.

(iii) Theta (Θ) :- This Notation describes both tight upper bound and tight lower bound of an algorithm. In real scenario the algorithm not always run on best and worst cases the avg running lies between best and worst and can be represent by 'Θ' Notation.

$$c_1 g(n) \leq f(n) \leq c_2 g(n). \qquad \forall\ n \geq \max(n_1, n_2)$$
$$c_1 \geq 0 \text{ and } c_2 \geq 0$$

Ques. 10:- For the function n^k and c^n, what is the asymptotic Relationship b/w these function ?

assume that $k >= 1$ of $c > 1$ are constants find out the value of c. and no. of which relationship hold.?

↳ as given n^k and c^n

Relationship between n^k and c^n is.

$$n^k = O(c^n)$$
$$n^k \leq a(c^n)$$
$$\forall \quad n \geq n_0 \quad \text{constant} \quad a > 0$$

for $n_0 = 1$: $c = 2$

$1^k < a^2$

$\Rightarrow \quad n_0 = 1$ and $c = 2$

Date : Page :
Topic :

Ques→2 For (i = 1 to n)
{ i = i * 2 }
i → 1, 2, 2^2, 2^3, 2^4, ..., 2^K.

$2^K = n$

take $\log_2$ both side.

$\log_2 2^K = \log n$

$K \log_2 2 = \log n$

$\boxed{K = O(\log n)}$

Ques→3 $T(n) = 3T(n-1)$ $n>0$, otherwise 1.

$T(n) = 3T(n-1)$ — (1)

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

From equ (1) :→

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

$= 3^2 T(n-2)$ — (2)

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

From equ (2) :→

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

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

⋮

$T(n) = 3^K T(n-K)$. — (3)

$n - K = 0$

$n = K$

STUDENT STYLE

Signature.....

Date _____ Page _____
Topic _____

From equ (3) :-

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

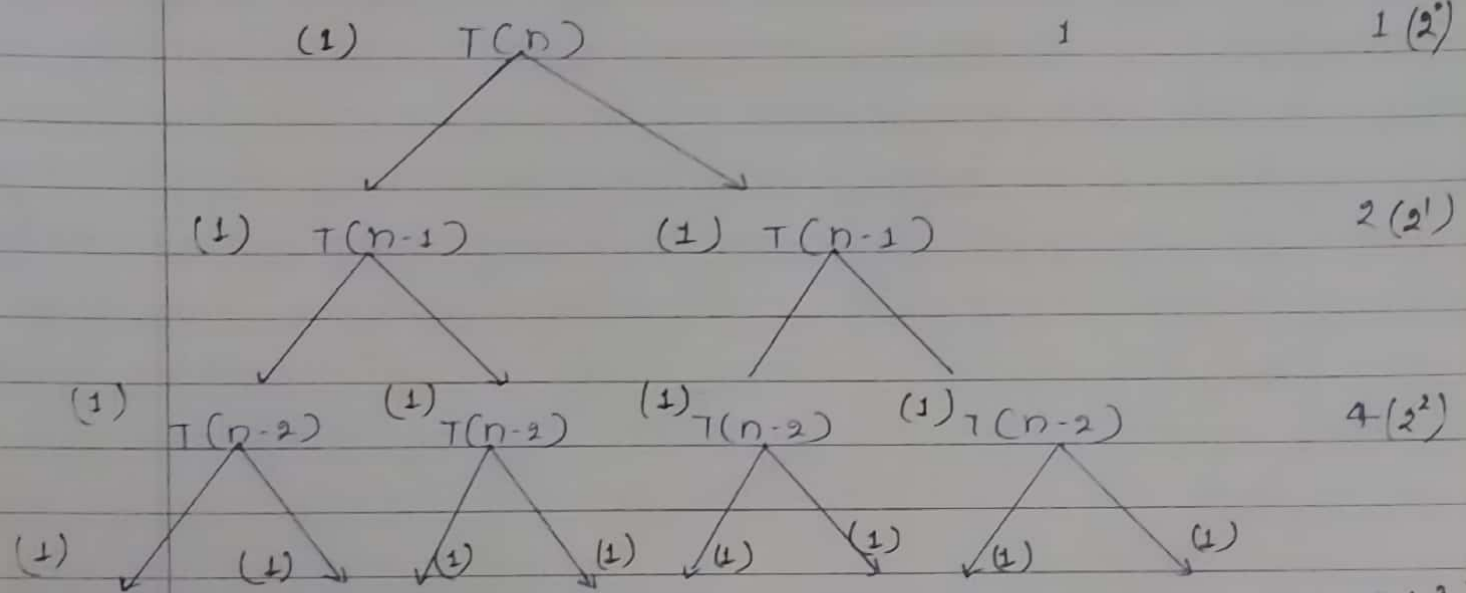
$= 3^k T(0)$

$T(0) = 1.$

$T(n) = 3^k$

$\boxed{T(n) = O(3^k)}$

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



$2^0, 2^1, 2^2, 2^3 \ldots\ldots, 2^k$

$T(n) = T(n-K) * 2^k$

$n - K = 0$

$n = K$

$T(n) = T(0) \cdot 2^k$

$\boxed{T(n) = O(2^n)}$

STUDENT STYLE

Signature _____

Date Page
Topic

Ques→5
```
int i=1, S=1;
while (S<=n){
    i++;
    S=S+i;
    Print+("#");
}.
```

S (Value of S):→

1, 3, 6, 10, ... n

(1) (1+2) (1+2+3) (1+2+3+4) (1+2+3+ ... +K)

Termination Condition.

$(1+2+3+\cdots+K) = n$

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

$K^2 + K = 2n$

$K^2 = n$

$K = \sqrt{n}$

$$\boxed{T(n) = O(\sqrt{n}).}$$

Ques→6.
```
void Function (int n){
  int i, count=0;
  for (i=1; i*i<=n; i++)
        count++;
}.
```

i→ 1, 2, 3, ..., K^2

$K^2 = n$ (Terminate the program)

$K = \sqrt{n}$ $K^2 > n$

$T(n) = O(\sqrt{n})$

STUDENT STYLE

Signature

Date : Page :
Topic :

Ques-7

```
void fun(int n) {
int i, j, k, count = 0;
for(i = n/2; i <= n; i++) {          loop 1.
    for(j = 1; j <= n; j = j * 2;) {   loop 2
        for(k = 1; k <= n; k = k * 2) {   loop 3
            count++;
        }
}.
```

loop 1 :→

$i \to \frac{n}{2}, \frac{n}{2}+1, \frac{n}{2}+2, \frac{n}{2}+3 \cdots \frac{n}{2}+k$

↳ $k = \frac{n}{2}$

$k > n$ (terminate pro).

$k = n$

$T(n) = O(n)$

For lop 1 and loop 2 we know time complexity.

$T(n) = \log n$.

total time complexity.

$T(n) = n * \log n * \log n$

$$\boxed{T(n) = O(n \log^2 n)}$$

STUDENT STYLE

Signature.....

Date : Page :
Topic :

Ques→8

```
fun (int n){
  if(n==0)
      return;
  for (i=1 to n)
      for (j=1 to n)
        print("#");

  fun (n-3)
}.
```

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

$= T(n-3) + n^2$ —— (1)

$T(n-3) = T(n-6) + (n-3)^2$

From equ (1) :→

$T(n) = T(n-6) + (n-3)^2 + n^2$

$T(n-6) = T(n-9) + (n-6)^2$

$T(n) = T(n-9) + (n-6)^2 + (n-3)^2 + n^2$

$T(n) = T(n-3K) + (n-(3K-3))^2 + [n-(3K-6)]^2 + n^2$

$n - 3K = 0$

$n = 3K$

$K = n/3$

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

$= T(0) + \frac{n(n+1)(n+2)}{6}$

$T(n) = n * n * n$

$\boxed{T(n) = O(n^3)}$

STUDENT STYLE

Signature.....

Date : Page :
Topic :

Ques 9. Void Fun(int n) {
  For (i=1 to n)
    for (j=1 ; j<=n ; j=j+i)
      printf("*");
}

i=01, j → 1, 2, 3, 4, ... , n
i=2, j → 1, 3, 5, 7, ... , n/2
i=3, j → 1, 4, 7, 19, ... , n/3
⋮
i=n, j → 1, ... , n/n

$$T(n) = n + \frac{n}{2} + \frac{n}{3} + \frac{n}{4} + \cdots + \frac{n}{n}$$

$$= n\left(1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n}\right)$$

→ logn.

$$T(n) = n \log n$$

$$\boxed{T(n) = O(\log n)}$$

STUDENT STYLE

Signature.....