Name- Prajjuwal Mehta
Sec - D
Rollno - 27

Date. ____
Page No. ____

Data Analysis of Algorithms

Q1

A1 These notations are used to tell the complexity of an algorithm when input is very large.

1 Big-O (O)

$$f(n) = O(C g(n))$$
iff
$$f(n) \le c\, g(n)$$
$\forall$ $n \ge n_0$ &
some constant, $c > 0$

2 Big Omega (Ω)
$$f(n) = \Omega(g(n))$$
$g(n)$ is "tight" lower bound

$$f(n) = \Omega(g(n))$$
iff
$$f(n) \ge c \cdot g(n)$$
$\forall$ $n \ge n_0$ &
some constant, $c > 0$

3 Theta (Θ)

$f(n) = \Theta(g(n))$

g(n) is both "tight" upper & lower bound of f(n).

$f(n) = \Theta(g(n))$

iff $c_1 g(n) \leq f(n) \leq c_2 g(n)$

$\forall \; n \geq \max(n_1, n_2)$

& some constant $c_1 > 0$

4 Small o (o)

$f(n) = o(g(n))$

g(n) is upper bound of func f(n)

$f(n) = o(g(n))$

when

$f(n) < c \cdot g(n)$

$\forall \; n \geq n_0$

and $\forall$ constant, $c > 0$

5 Small omega (ω)

$f(n) = \omega(g(n))$

g(n) is lower bound of func f(n)

$f(n) = \omega(g(n))$

when $f(n) > c \cdot g(n)$

$\forall \; n > n_0$ and $\forall \; c > 0$

Q2

for (i = 1 to n) // i = 1, 2, 4, 8, ... n
{ i = i * 2 } // O(1)

$\Rightarrow \sum_{\substack{i=1 \\ (i=i*2)}}^{n} 1+2+4+8+ \cdots +n$

k^{th} term of GP $\Rightarrow T_k = ar^{k-1}$

$n = 1 * 2^{k-1}$

$n = 2^{k-1}$

$n = \frac{2^k}{2}$

$2n = 2^k$

$\log_2 2n = k \log_2 2$

$\log_2 2n = k$

$k = \log_2 2 + \log_2 n$

$k = 1 + \log_2 n$

$O(\log_2 n)$

Q3 $T(n) = \{ 3T(n-1)$ if $n>0$, otherwise

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

put $n = n-1$ in (1)

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

put $T(n-1)$ in (1)

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

$$T(n) = 9T(n-2) \quad -(3)$$

put $n = n-2$ in (1)

$$T(n-2) = 3T(n-3) - (4)$$

put in (3)

$$T(n) = 27T(n-3) - (5)$$

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

let $n-k=1$

$n-k=1$

$k=n-1$

$$T(n) = 3^{n-1} T(n-n+1)$$

let $k=n$

$$T(n) = 3^n T(0)$$

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

complexity $= O(3^n)$

Q.4 $T(n) = \{2T(n-1) - 1$ if $n > 0$, otherwise
$1\}$

$$T(n) = 2T(n-1) - 1 \quad — (1)$$

put $n = n-1$ in (1)

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

put $T(n-1)$ from (2) to (1)

$$T(n) = 2\cancel{T}(2T(n-2) - 1) - 1$$

$$T(n) = 4T(n-2) - 3 \quad - (3)$$

put $n = n-2$ in (1)

$$T(n-2) = 2T(n-3) - 1 \quad - (4)$$

put $T(n-2)$ from (4) to (3)

$$T(n) = 4(2T(n-3) - 1) - 3$$

$$T(n) = 8T(n-3) - 7 \quad - (5)$$

$$T(n) = 2^k T(n-k) - 2^k - 1 \quad - (6)$$

let $n-k = 1$ $\quad k = n-1$

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

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

$$T(n) = 2^k T(1) - 2^{n-1} - 1$$

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

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

Q5

```
int i = 1, s = 1;
while (s <= n)
{
    i++;
    s = s + i;
    printf("#");
}
```

$i = 2, 3, 4, 5, 6 \ldots K$

$s = 3, 6, 10, 15, \ldots K$

when $s >= n$, then loop will stop

K^{th} iteration

$\rightarrow 2 + 2 + 3 + 4 + \ldots + K = n$

$= 1 + (K * (K+1))/2 = n$

$K^2 = n$

$K = \sqrt{n}$

$= O(\sqrt{n})$

Q.6

```
void function (int n)
{
    int i, count = 0;
    for (i=1; i*i<=n; i++)
        count ++;
}
```

as $i^2 < n$

$i < \sqrt{n}$

$i = 1, 2, 3, 4 \ldots \sqrt{n}$

$\sum_{i=1}^{\sqrt{n}} 1+2+3 \cdots + \sqrt{n}$

$T(n) = \frac{\sqrt{n} \times (\sqrt{n}+1)}{2} = \frac{n+\sqrt{n}}{2}$

$T(n) = O(n)$

Q7

```
void function(int n)
{
    int i, j, k, count = 0;
    for (i = n/2; i <= n; i++)  - o(logn)
      for (j = 1; j <= n; j = j*2)  - o(logn)
        for (k = 1; k <= n; k = k*2)
            count++
    }
    O(n log^2 n)
```

Q8

```
function (int n)
{
    if (n==1)
      return;
    for (i=1 to n)
    {
        for (j=1 to n)
        {
            printf ("*");
        }
    }
    function(n-3);
}
```

Q9

```
void function(int n)
{
    for(i=1 to n)
    {
        for(j=1; j<=n; j=j+i)
            printf("*");
    }
}
```

$i=1$, $j=1,2,3,4,5,6,\ldots n$

$i=2$, $j=1,3,5,7,9\ldots \; n/2$

$i=3$, $j=1,4,7,10,13 \ldots \; n/3$

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

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

$$= n \log n$$

$O(n \log n)$

Q10

as given n^k & c^n

relation between n^n & c^n is

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

as $n^k \le c^n$

$\forall\ n \ge n_0$ & some constant

for $n_0 = 1$

$c = 2$

$\Rightarrow 1^k \le a2^1$

$\Rightarrow n_0 = 1$ & $c = 2$