Page No.
Date

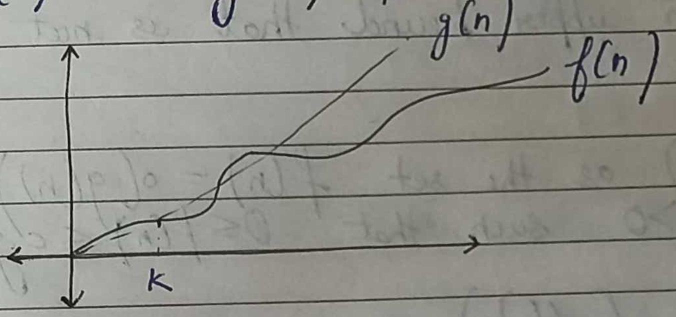
ASSIGNMENT - 1

Q. What do you understand by Asymptotic notations. Define different Asymptotic notation with examples.
Asymptotic notations are mathematical tools used to describe the behaviour of functions as their input grows towards infinity.

→ There are 5 asymptotic notations

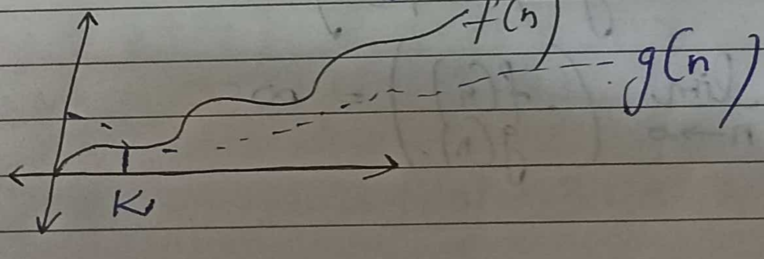
1. Big O notations (O):
   - express the upper bound of the algorithm's running time. (asymptotically tight)
   - measures the worst case time complexity

$f(n) \leq c \cdot g(n)$ for all case $n > n_0$



2. Big Omega Ω:
   - express the lower bound of the algorithm's running time, (asymptotically tight)
   - measures the best case time complexity

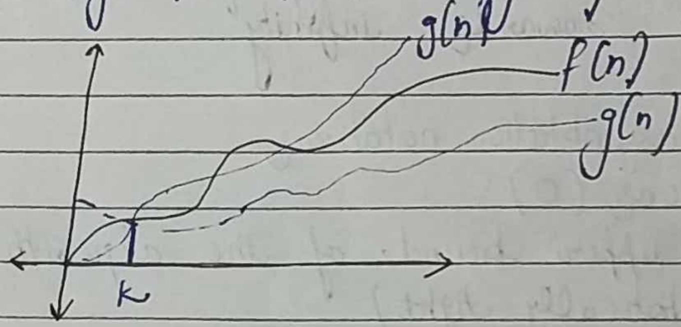
$f(n) = \Omega(g(n))$ when there exist constant c that $f(n) \geq c \cdot g(n)$



Page No.
Date

3 θ : Theta
express both lower bound and the upper bound of an algorithm's running time

$f(n) = \theta(g(n))$ $\forall$ c_1 & c_2
$c_1 \cdot g(n) \le f(n) \le c \cdot g(n)$



4. little Oh , o
denote an upper bound that is not asymptotically tight

for $o(g(n))$ as the set $f(n) = o(g(n))$
$c > 0$ such that $0 \le f(n) \le c \cdot g(n)$

$$\lim_{n\to\infty} \left(\frac{f(n)}{g(n)}\right) = 0$$

5. little Omega , ω
denote lower bound that is asymptotically tight
$\omega(g(n))$ as the set $f(n) = \omega(g(n))$
$c > 0$ and there exist a value $n_0 > 0$
$0 \le c \cdot g(n) < f(n)$

$$\lim_{n\to\infty} \left(\frac{f(n)}{g(n)}\right) = \infty$$

Page No.
Date

Q.2 What should be the time complexity of

```
for(i-1 to n)
{    i=i*2}
```

Iteration let $i = 1$

1	$i = 1*2 = 2$
2.	$i = 2*2 = 4$
3	$i = 4*2 = 8$

limit
$= n$

Complexity = $\log(n)$

Q.3 $T(n) = \{3T(n-1)$ if $n>0$ otherwise $1\}$

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

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

$\vdots$

$T(n-3) = 3T(n-4)$
$T(n-4) = 3T(n-5)$

$\vdots$

$T(1) = 3T(0) = 3(1) = 3$
$T(2) = 3T(1) = 3(3)$

$T(n) = 3.3. \ldots 3T(0) = 3.3. \ldots 3(1)$

Page No. ______
Date

-3^n

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

$T(n-1) = 2T((n-1)-1-1)$
$= 2T(n-2)-1$
$T(n-2) = 2T(n-2)-1-1$
$= 2T(n-3)-1$
$\vdots$
$T(1) = 2T(0)-1 \quad \Rightarrow T(0)=1$
$= 2(1)-1$
$= 1$
$T(n) = 2(2(2(\ldots 2(T(0)-1)-1)\ldots -1)$
$T(n) = 2^n \cdot T(0) - (2^n - 1)$
$= 2^n - (2^n - 1)$
$= 1$

Q.5

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

Iteration
1 $\quad i = 2$
$\quad s = 3$
2 $\quad i = 3$
$\quad s = 6$
$s = 1, 3, 6, 10 \ldots\ldots$

Page No.
Date

last term = k

$$\text{Sum} = \frac{k}{2}(1+k) \le n$$

$$\frac{k}{2}(1+k) \le n$$

$$k + 2k \le 2n$$

$$\Rightarrow \quad k = \frac{-1+\sqrt{1+8n}}{2}$$

the complexity will be approx k
hence ignoring constant $(O\sqrt{n})$

Q.6
```
void funtion (int n)
{
        int count = 0;
        for( i = 1 ; i*i <= n ; i++ )
        {
                c++;
        }
}
```

loop $i * i \le n$
$i^2 \le n$
$i \le \sqrt{n}$

hence loop will continue till $\sqrt{n}$
$O(\sqrt{n})$

Q.7.
```
void function (int n)
{ int i, j, count = 0;
  for ( i = n/2 ; i <= n, i++ )
        for( j = 1 ; j <= n ; j = j*2 )
                for( k = 1 ; k <= n ; k = k*2 )
                        count ++
```

Page No. ______
Date

$i = n/2$ to n
$j = 1$ to n
$k = 1$ to n

for $i = n/2$ to n no of times $= n/2 \approx n$

for $j = 1$ to n no. of times $= \log n$
{ $j = j*2$ }
for $k = 1$ to n no. of times $= \log n$
{ $j = j*2$ }
removing constants $= O(n \log^2 n)$

Q.8
```
function (int n)                          T(n)
{
    if (n == 1)
      return;                              O(1)
    for (j = 1 to n)                       { O(n^2) → 2 nested
    {                                        O(n^2)      loop }
        print("*");
    } function (n-3);                      T(n-3)
}
```
one more loop of i

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

hence approx $O(n^2)$

Q.9
```
void function (int n)
{       for (i=1 to n)
```

Page No. ____
Date

```
{
    for (i. 1 to n)
        for (j to ≤ n ; j = j + i)
            print ("");
        }
    }
}
```

for (1 = 1 to n) $\longrightarrow O(n)$

for (j to ≤ n ; j = j + i) n/i = times

$j = \{(1-n)\ (1, 2, .. 3)\}$ (for i = 1) $\{j = 1, 3, 5 \ldots\}$ (for i = 2)

$\rightarrow n \cdot \left(1 + \frac{1}{2} + \frac{1}{3} + \ldots 1/n\right\} = O(\log n)$

Time complexity $O(n \log n)$

Q.8. { Correct answer }

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

<u>Ans.</u>

Page No. ______
Date

Q.10

n ^ k and c ^ n

Assume k >= 1 and c > 1. c = ?

n_0 = ?

for n ^ k growth rate $\propto$ k

for c^n growth rate $\propto$ c

- Set n = 2

  n = 2

  $\therefore n^k = 2^k$

  $\cdot c^n = 2^2$

  for $c^2 > 2^k$

  $c^2 > 2^k$

- Set n = 3

  k = 1

  $3^2 = 9$

  $2^1 = 2$

  9 > 2 c^n grows faster than n^k

  n = 2

relationship $c^n > n^k$ holds for large values of n