Name: Nikita kumari Section: CST SPL-1
Roll: 19 Date: 10/3/22

Design, Analysis of Algorithms

Tutorial - 1

Q1. Asymptotic Notations :- It is the mathematical way of representing the time complexity. It is used to describe the running time of an algorithm, how much time an algorithm takes with a given input, n.

→ There are mainly three different Asymptotic Notations:

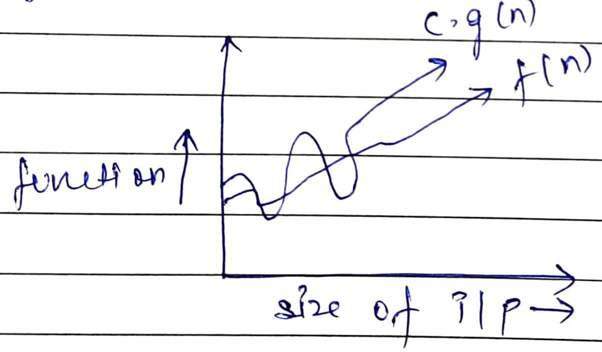
1. Big - O (O) (worst case)

⇒ $f(n) = O\ g(n)$

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

$\forall$ $n \geq n_0$, for some constant, $c > 0$

⇒ $g(n)$ is "tight" upper bound of $f(n)$.



2. Big. Omega (Ω) (Best case)

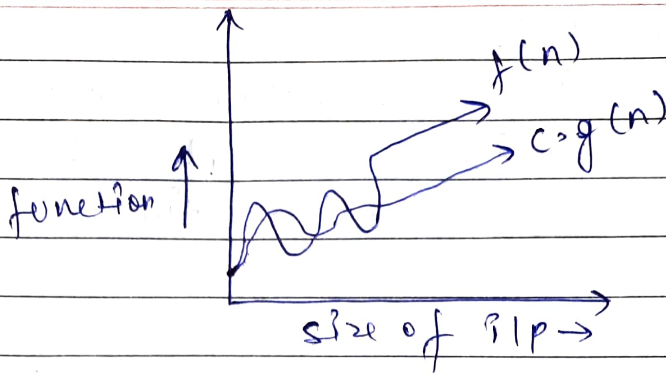
⇒ $f(n) = \Omega(g(n))$

$g(n)$ is "tight" lower bound of function $f(n)$

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

iff $f(n) \geq c \cdot g(n)$

$\forall$ $n \geq n_0$, for some constant, $c > 0$.



3. Theta (θ)

⇒ $f(n) = \theta\, g(n)$

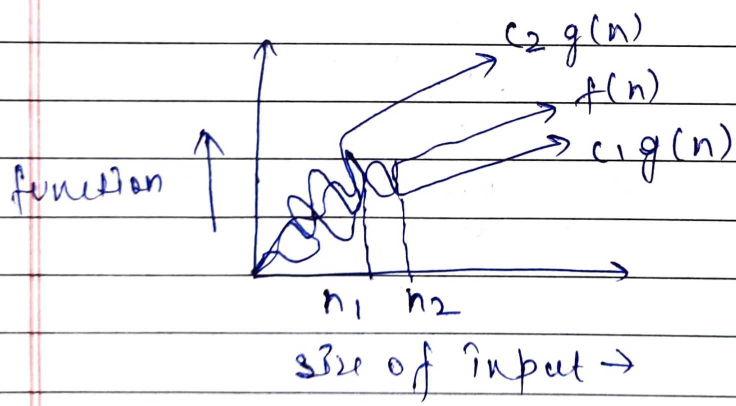
g(n) is both "tight" upper and lower bound of function f(n).

$$f(n) = \theta\, g(n)$$

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

$\forall\ n \ge \max(n_1\ n_2)$

for some constant $c_1 > 0$ & $c_2 > 0$.



Examples :-

1. Big O.

→
```
#include<stdio.h>
int main(){
for(int i=0; i<=n; i++)          ⇒   O(n)
{
// int sum=sum+i;
}
}
```

Date :
Page :

2. Big Omega (Ω)

→ The worst case running time of binary search is $\Omega(1)$, because it takes at least constant time.

3. Theta (θ)

→ The high temperature today will be 20° C and the low will be 10° C.

Q2.

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

⇒ n = 1   i = 1 times

n = 10   i = 4 times

⋮   ⋮

n times $(\log_2 n+1)$ times

⇒ $\boxed{T_n = O(\log_2 n)}$ Ans

Date :
Page :

Q3. $T(n) = 3T(n-1) \quad , \quad n > 0$
$T(0) = 1$

$\Rightarrow \quad T(n) = 3T(n-1) \quad — (i)$

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

$T(n-1) = 3T(n-2) \quad — (ii)$

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

$T(n-2) = 3T(n-3) \quad — (iii)$

putting (ii) in (i)

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

putting (iii) in (ii)

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

- - - - - - - - - - - -

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

putting $n-k = 0$; $n = k$.

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

$= 3^n \cdot T(0) = 3^n(1)$

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

Q4. $T(n) = 2T(n-1) - 1$
$T(0) = 1$

$\Rightarrow T(n) = 2T(n-1) - 1 \quad — (i)$

putting $n = n-1$ in (i)

$T(n-1) = 2T(n-2) - 1 \quad — (ii)$

putting $n = n-2$ in (i)

$T(n-2) = 2T(n-3) - 1 \quad — (iii)$

putting (ii) in (i)

$= \quad T(n) = 2\left(2\,T(n-2)-1\right)-1$

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

$= \quad 2^2\,T(n-2) - 3 \quad — \text{(iv)}$

putting (iii) in (iv)

$T(n) = 2^2\left(2\,T(n-3)-1\right) - 3$

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

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

$- - - - - - - - - -$

$T(n) = 2^k\,T(n-k) - 2^{k-1} - 2^{k-2} \ldots 2^0$

putting $n-k=0$; $n=k$

$\Rightarrow T(n) = 2^n\,T(0) - 2^{n-1} - 2^{n-2} \ldots 2^0$

$= 2^n - 2^{n-1} - 2^{n-2} \ldots 2^0$

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

$= \cancel{2^n} - \cancel{2^n} + 1$

$\boxed{T(n) = O(1)}$ Ans.

Q7.

$i \qquad j \qquad k$

$\frac{n}{2} \qquad (\log_2 n) \qquad (\log_2 n)$

$n \qquad \log_2 n \qquad \log_2 n$

$= \left(\frac{n}{2}+1\right)$ times $\quad (\log_2 n)$ times $\quad (\log_2 n)$ times

$O(i * j * k) = O\left(\left(\frac{n}{2}+1\right) * \left(\log_2 n\right) * \left(\log_2 n\right)\right)$

$= O\left(n\,(\log n)^2\right)$ Ans.

Q5.

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

$$\sum_{s=1}^{n} 1+1+1+ \cdots \sqrt{n}$$

$\therefore$ $T(n) = O(n^{1/2})$ Ans.

Q6.

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

~~n = 8 1+1+...~~

$$\sum_{i=1}^{n} 1+1+1 \cdots \sqrt{n} \text{ times}$$

$\therefore$ $O(\sqrt{n})$ or $O(n^{1/2})$. Ans.

Q8. $T(n) = T(n-3) + n^2$ — (I)

$T(1) = 1$

put $n = n-3$ in (I)

$T(n-3) = T(n-6) + (n-3)^2$ — (II)

put $n = n-6$ in (I)

$T(n-6) = T(n-9) + (n-6)^2$ — (III)

putting (II) in (I)

$\Rightarrow T(n) = T(n-6) + (n-3)^2 + n^2$ — (IV)

putting $T(n-6)$ in (IV)

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

$= T(n-3\cdot3) + (n-3\cdot2)^2 + (n-3\cdot1)^2 + (n-3\cdot0)^2$

— — — — — — — — —

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

$- - - - n^2$

putting $n - 3K = 1$; $n = 3K+1$

$K = \frac{n-1}{3}$

$T(n) = T(1) + (1+3)^2 + (1+6)^2 - - - n^2$

$= 1 + 4^2 + - - - n^2$

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

Q9.

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

for $i = 1$, $j = 1, 2, 3, \ldots n$
for $i = 2$, $j = 1, 3, 5, \ldots n/2$
for $i = 3$, $j = 1, 4, 7, \ldots n/3$

$i = n$, $j = 1 + 1 + \ldots 1$.

$= \frac{n}{1} + \frac{n}{2} + \frac{n}{3} + \ldots \frac{n}{n-1} - \log(n-1)$

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

$= n\log(n-1) - \log(n-1)$

$= n\log(n-1)$

$\boxed{T(n) = n\log n}$ Ans.