Tutorial 1

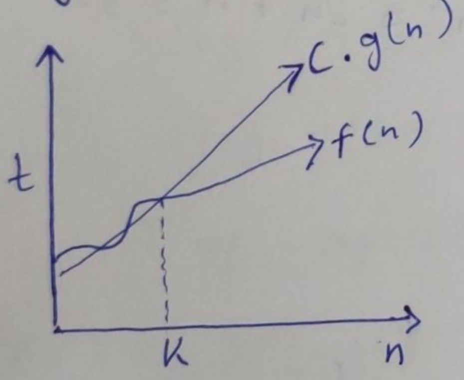
Name - Tanvi Nauthyal
Section - G
Roll No. - 47

Q. 1) Asymptotic Notations are mathematical tools to represent the time complexity of algorithms for asymptotic analysis.

- These notations are used to tell the complexity of an algorithm when the input is very large (towards infinity).

Types of Asymptotic Notations :

i) Big - Oh (O)



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

$f(n) \leq c.\,g(n)$

$c > 0$

$n \geq K$

$K \geq 0$

Eg. $f(n) = 2n^2 + n$

$2n^2 + n \leq c.\,g(n^2)$

$2n^2 + n \leq 3n^2$

$n \leq n^2$

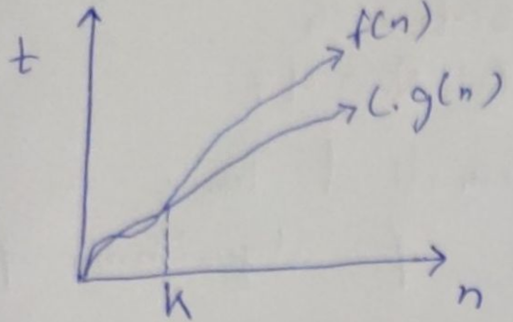
$1 \leq n$

$n \geq 1$

→ gives least upper bound

→ Worst case

ii) Big-Omega (Ω)



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

$f(n) \geq c.g(n)$

$2n^2 + n \geq c.n^2$

$c = 2$

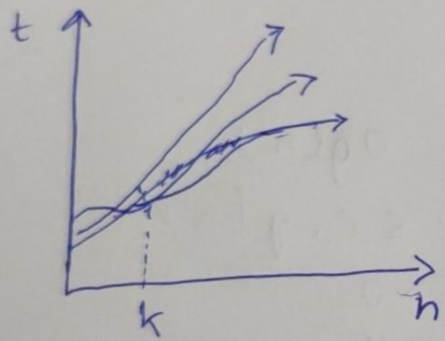
$2n^2 + n \geq 2.n^2$

$n \geq 0$

→ Greatest lower bound

→ Best case

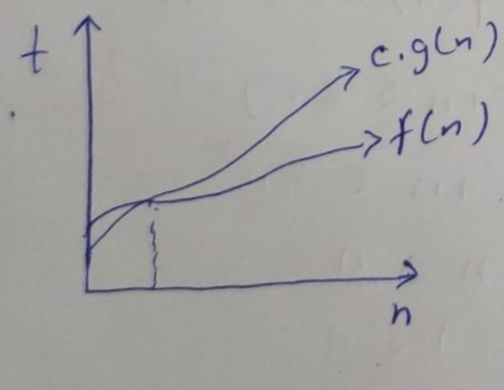
iii) Theta (θ)



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

$2n^2 \leq 2n^2 + n \leq 3n^2$

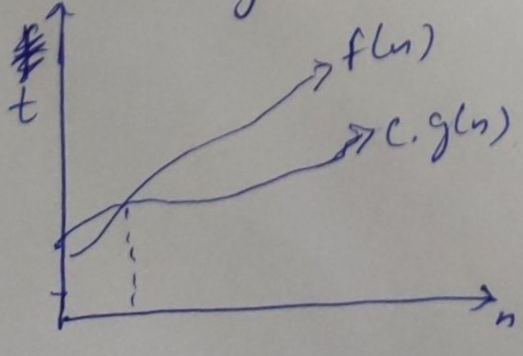
iv) Small-oh (o)



$c.g(n) > f(n)$

$n^3 > n^2$

v) Small-omega (ω)



$f(n) > c.g(n)$

$n^2 > n$

Q.2) for (i=1 to n)
{ $i = i * 2;$ $\rightarrow O(1)$
}

for $i = 1, 2, 4, 6, 8 \ldots n$ times
i.e series is in GP
So $a = 1$, $r = 2/1$

K^{th} value of GP :

$t_K = a\, r^{K-1}$

$t_K = 1(2)^{K-1}$

$2n = 2^K$

$\log_2(2n) = K \log 2$

$\log_2 2 + \log_2 n = K$

$\log_2 n + 1 = K$

$\therefore$ Time complexity $\Rightarrow O(\log n)$ Ans.

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

$T(n) = 3T(n-1)$ — ①
$T(n) = 1$
put $n = n-1$ in ①
$T(n-1) = 3T(n-2)$ — ②
put ② in ①
$T(n) = 3 \times 3T(n-2)$
$T(n) = 9T(n-2)$ — ③
Put $n = n-2$ in ①
$T(n-2) = 3T(n-3)$
put in ③
$T(n) = 27T(n-3)$ — ④

Generating Series :

$T(k) = 3^k T(n-k)$ — (5)

for k^{th} terms, let $n-k = 1$ (Base Case)

$k = n-1$

put in (5)

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

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

$T(n) = O(3^n)$ Ans.

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

$T(n) = 2T(n-1) - 1$ — (1)

put $n = n-1$

$T(n-1) = 2T(n-2) - 1$ — (2)

put in (1)

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

$= 4T(n-2) - 2 - 1$ — (3)

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

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

Put in (1)

$T(n) = 8T(n-3) - 4 - 2 - 1$ — (4)

Generating series

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

<u>k^{th} term</u>

let $n - k = 1$

$k = n - 1$

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

$$= 2^{k-1} - 2^{k-1}\left(\frac{1}{2} + \frac{1}{2^2} + \cdots \frac{1}{2^{k-1}}\right)$$

is series in GP

$a = 1/2$, $r = \frac{1}{2}$

so,

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

$$= 2^{k-1}\left(1 - 1 + (1/2)^{n-1}\right)$$

$$= \frac{2^{n-1}}{2^{n-1}}$$

$T(n) = O(1)$ Ans.

Q. 5) What should be the time complexity of

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

$i = 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad \ldots$

$s = 1 + 3 + 6 + 10 + 15 + \ldots$

Sum of $s = 1 + 3 + 6 + 10 + \ldots + n$ — (1)

Also $s = 1 + 3 + 6 + 10 + \ldots T_{n-1} + T_n$ — (2)

$0 = 1 + 2 + 3 + 4 + \ldots, n - T_n$

$T_k = 1 + 2 + 3 + 4 + \ldots + k$

$T_k = \frac{1}{2} k(k+1)$

for k iterations

$1 + 2 + 3 + \dots k <= n$

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

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

$$O(k^2) <= n$$

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

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

Q.6> Time complexity of

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

-> As $i^2 = n$

$i = \sqrt{n}$

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

$$\sum_{i=1} 1 + 2 + 3 + 4 + \dots + \sqrt{n}$$

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

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

$T(n) = O(n)$ Ans.

Q.7) Time Complexity of

```
void f (int n)
{
 int i, j, K, count = 0;
 for (int i = K/2 ; i <= n ; ++i)
  for (j = 1 ; j <= n ; j = j * 2)
   for (K = 1 ; K <= n ; K = K + 2)
      count ++;
}
```

$\because$ $K = K^2$

$K = 1, 2, 4, 8, \ldots n$

$\therefore$ series is in GP

So, $a = 1$, $r = 2$

$$\frac{a(r^K - 1)}{r - 1}$$

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

$n = 2^K - 1$

$n + 1 = 2^K$

$\log_2(n) = K$

i	j	K
1	log(n)	log(n) * log(n)
2	log(n)	log(n) * log(n)
⋮	⋮	⋮
n	log(n)	log(n) * log(n)

T.C $= O(n * \log n * \log n)$

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

Q.8) Time complexity of

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

for (i = 1 to n)

we get j = n times every turn

$\therefore i * j = n^2$

k^{th}, Now

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

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

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

and $T(1) = 1$

Now, substitute each value in $T(n)$

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

Let

$k^n - 3K = 1$

$K = (n-1)/3$

total times $= K + 1$

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

$T(n) \simeq Kn^2$

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

so,

$T(n) = O(n^3)$ Ans.

Q.9) Time Complexity of:

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

for $i = 1$ $\quad j = 1 + 2 + \ldots \ldots (n \geq j+i)$

$i = 2$ $\quad j = 1 + 3 + 5 + \ldots \ldots (n \geq j+i)$

$i = 3$ $\quad j = 1 + 4 + 7 + \ldots \ldots (n \geq j+i)$

n^{th} term of AP is

$T(n) = a + d * m$

$T(n) \; 1 + d * m$

$(k-1)/d = m$

for $i = 1$ $\quad (n-1)/1$ times

$i = 2$ $\quad (n-1)/2$ times

$\vdots$

$i = k-1$

we get,

$T(n) = i_1 j_1 + i_2 j_2 + \ldots, i_{n-1} j_{n-1}$

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

$= n + n/2 + n/3 + \ldots \; n/n-1 - n*1$

$= n \left[1 + 1/2 + 1/3 + \ldots \; 1/n-1\right] - n*1$

$= n \times \log n - n + 1$

Since $\int 1/n = \log n$

$T(n) = O(n \log n)$ Ans.

Q.10) For the function n^k & c^n, what is the asymptotic Relationship b/w these functions? Assume that $k >= 1$ & $c > 1$ are constants. find out the value of c & n_0, of which relationship holds.

As given n^k and c^n

Relationship b/w n^k & c^n is

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

$$n^k \leq a(c^k)$$

$\forall\ n \geq n_0$ & constant $a > 0$

for $n_0 = 1$; $c = 2$

$\Rightarrow 1^k < a^2$

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