Tutorial -1

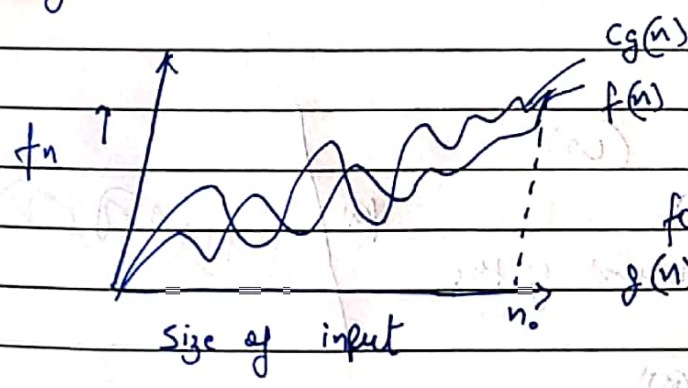
Name :- Divyansh Sethi
Section :- CST SpL2
Semester :- IV
class Roll No :- 06
University Roll no :- 2017491

Divyansh Sethi

Q1 Asymptotic Notation
↳ Tending to infinity

They help you find the complexity an algorithm when input is very large.

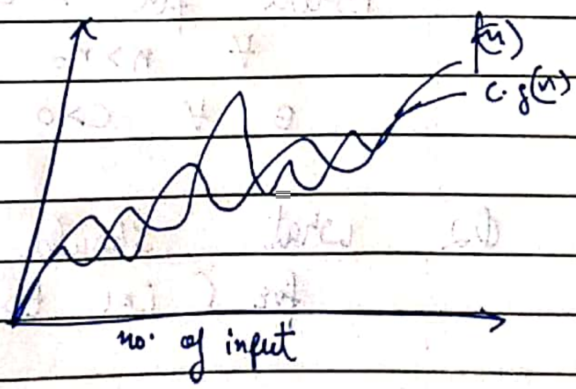
1) Big O (o)



$f(n) = O(g(n))$
iff $f(n) \le C.g(n)$
$\forall\ n \ge n_0$
for some constant $c>0$
g(n) is tight upper bound of f(n)

11) Big Omega (Ω)

$f(n) = \Omega(g(n))$
g(n) is tight lower bound of f(n)



$f(n) = \Omega(g(n))$
iff $f(n) \ge c.g(n)$
$\forall\ n \ge n_0$ for some constant $c>0$

iii) Theta (θ)

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

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

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

iff

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

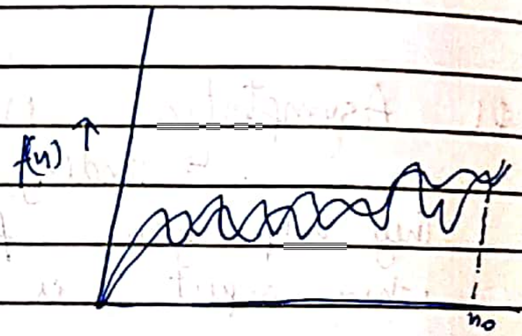
$\forall\ n \ge \max(n_1, n_2)$

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

4.) Small o (o)

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

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



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

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

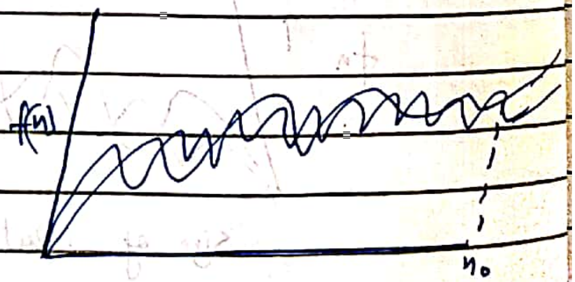
$\forall\ n > n_0$

& $\forall\ c > 0$

5) small omega (ω)

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

$g(n)$ is lower bound of the fn $f(n)$



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

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

$\forall\ n > n_0$

& $\forall\ c > 0$

Q2 what should be time complexity of

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

Divyansh

Date
Page No.

for $(i=1$ to $n)$ // $i=1, 2, 4, 8 \ldots n$
$(i = i*2)$ // $O(1)$

$\Rightarrow \sum_{i=1}^{n} \quad 1+2+4+8+\cdots+n$

GP kth value $\Rightarrow T_k = ar^{k-1}$

$1 \times 2^{k-1}$

$n = 2^{k-1}$

$2n = 2^k$

$\log 2n = k \log 2$

$\log_2 + \log n = k \log_2$

$\log n+1 = k$

$O(k) = O(1+\log n)$

$O(\log n)$

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

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

put $n = n-1$

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

from (2) & (1)

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

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

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

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

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

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

putting $n-k=0$

$n=k$

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

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

Divyansh

Date

Page No.

$T(0) = 1$

$T(n) = 3^n \times 1$

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

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

$T(n) = 2T(n-1) - 1$ — ①

Let $n = n-1$

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

from ① & ②

$T(n) = 2[2T(n-2) - 1]$ — ③

Let $n = n-2$

$T(n-2) = 2T(n-3) - 1$ — ④

③ & ④

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

$T(n) = 8T(n-3) - 4 - 2 - 1$

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

$GP = 2^{k-1} + 2^{k-2} + 2^{k-3} + \dots 1$

$a = 2^{k-1}$

$r = 1/2$

$= \dfrac{a(1-r^n)}{1-r}$

$= \dfrac{2^{k-1}(1-(1/2)^n)}{1/2}$

$= 2^k(1-(1/2)^k)$

$2^k - 1$

Let $n-k = 0$

$n = k$

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

$T(n) = 2^n \cdot 1 - (2^n - 1)$

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

$T(n)$ $O(1)$

Divyansh

Date
Page No.

Q5 what should be time complexity of

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

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

$s : 1 + 3 + 6 + 10 + 15 + 21 \ldots \; n$

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

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

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

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

for k iteration

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

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

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

$O(k^2) <= n$

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

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

Q6 Time Complexity of

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

Dinesh

Date
Page No.

as $i^2 <= n$

$i <= \sqrt{n}$

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

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

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

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

$$T(n) = O(n)$$

Q7 Time Complexity of

```
void fn (int n)
{ int i, j, k, 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 ++ ;
}
```

for $k = k*2$

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

GP $a=1$ $r=2$

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

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

$n \Rightarrow 2^k$

$\log n \Rightarrow k$

Date
Page No.

i	j	k
1	log n	log n * log n
2	log n	log n * log n
:	:	:
n	log n	log n * log n

$O(n * \log n * \log n)$

$O(n \log^2 n)$

Q8 Time complexity of

```
function (int n)
{ int (n==1)
    return;                     // O(1)
  for (i=1 to n)                // i = 1, 2, 3, 4 ... n   O(n)
  { for (j=1 to n)
     { print ('*');
     }
  }
}
function (n-3);                 T(n/3)
```

→ $T(n) = T(n/3) + n^2$

$a = 1 \quad b = 3 \quad f(n) = n^2$

$c = \log_3 1 = 0$

$n^0 = 1 > f(n) = n^2$

$T(n) = \Theta(n^2)$

Q9 Time complexity of

void function (int n)

Divyansh

Date

Page No.

```
{ for (i=1 to n)
  { for (j=1; ~~for j=n~~ j<=n; j=j+i)
      print ("*");
  }
}
```

for $i=1$, $j = 1, 2, 3, 4 \dots n$ $\quad n$

for $i=2$, $j = 1, 3, 5 \dots n$ $\quad n/2$

for $i=3$ $j = 1, 4, 7 \dots n$ $\quad n/3$

for $i=n$ $j=1$ $\quad 1$

$$\sum_{j=n}^{1} n + n/2 + \frac{n}{3} + \frac{n}{4} + \dots + 1$$

$$\sum_{j=n}^{1} n\left[1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots \frac{1}{n}\right]$$

$$\sum_{j=n}^{1} n[\log n]$$

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

Q10 for function n^k c^n what is the asympto relation b/w these functions?
assume that $k >= 1$ & $c > 1$ are constant
find out the value of c & n_0 for which relation holds

as given n^k & c^k
relation b/w n^k & c^n is

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

as $n^k \leq a c^n$

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

for $n_0 = 1$

$c = 2$

$1^k \le a\,2^1$

$n_0 = 1$ & $c = 2$

Dilyan