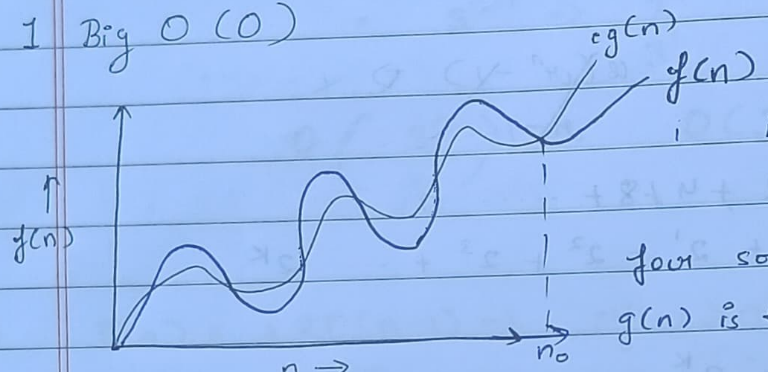
Name - Yash Singhal
Section - CST spl 2
Roll no. - 25

classmate
Date
Page

Q1 Asymptatic Notations.

Asymptatic means tending to infinity. They help us find the complexity of an algorithm when input is very large.

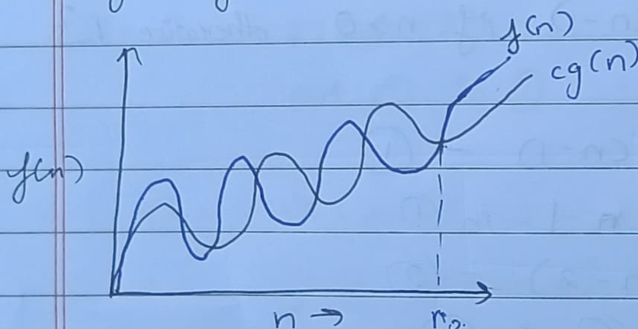
1 Big O (O)




$f(n) = O(g(n))$
iff $f(n) \leq cg(n)$
$\forall n \geq n_0$

for some constant $c>0$
g(n) is tight upper bound of f(n)

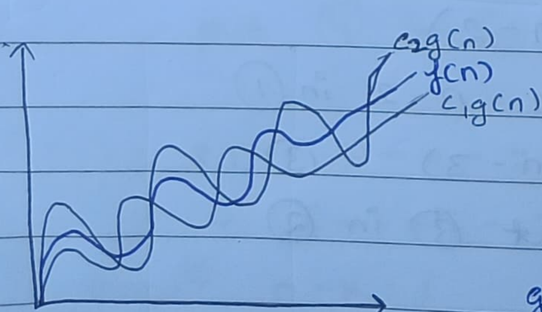
2 Big Omega ($\Omega$)




$f(n) = \Omega(g(n))$
iff $f(n) \geq c.g(n)$
$\forall n \geq n_0$.

g(n) is tight Lower bound of f(n)

3 Theta ($\Theta$)




$f(n) = \Theta(g(n))$
iff $c_1g(n) \leq f(n) \leq c_2g(n)$
$\forall n \geq max(n_1, n_2)$

g(n) is both "tight upper & Lower bound" of function f(n)

classmate

Date

Page

Q2 for ( $i = 1$ to $n$ ) // $i = 1, 2, 4 \ldots n$.

$i = i * 2$ // $O(1)$

$$\sum_{i=1}^{n} 1 + 2 + 4 + 8 \ldots n$$

$a = 1, r = 2$

G.P with values, $T_k = a r^{k-1}$

$$\cancel{\frac{a(r^n - 1)}{r - 1}} \quad ? \times$$

$i = 1 + 2 + 4 + 8 + \ldots n$

$= 2^0 + 2^1 + 2^2 + 2^3 + \ldots 2^k$

$\Rightarrow n = 2^k$

$\Rightarrow \log_2 n = k \log_2 2$

$\Rightarrow k = \log_2 n \quad = O(\log n)$

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

$T(n) = 3T(n-1)$ — (1)

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

$T(n-1) = 3T(n-2)$ — (2)

put (2) in (1)

$T(n) = 9T(n-2)$

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

$T(n-2) = 3T(n-3)$ — (3)

put (3) in (2)

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

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

$\Rightarrow n - K = 1$

$K = n - 1$

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

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

$= 3^{n-1} \Rightarrow \frac{3^n}{3^1}$

$O\left(\frac{3^n}{\cancel{3^1}}\right) \Rightarrow O(3^n)$

Q4 $T(n) = \{2T(n-1) - 1$, if $n > 0$, otherwise $1\}$

$T(n) = 2T(n-1) - 1$ — (1)

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

$T(n-1) = 2T(n-2) - 1$ — (2)

put (2) in (1)

$T(n) = 4T(n-2) - 2 - 1$ — (3)

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

$T(n-2) = 2T(n-3) - 1$ — (4)

put (4) in (3).

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

$= 2^K T(n-K) - 2^{K-1} - 2^{K-2} - \dots +$

$\Rightarrow n - K = 1$

$n - 1 = K$

$$T(n) = 2^{n-1} T(n-n+1) - 2^{n-2} - 2^{n-3} - 2^{n-4} \ldots 1$$

$$= \frac{2^n}{2} - \frac{2^n}{2^2} - \frac{2^n}{2^3} - \frac{2^n}{2^4} \ldots 1$$

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

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

Sum of G.P $\Rightarrow \frac{a(1 - r^{term})}{1 - r}$

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

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

$$= O(2^n - 1) \Rightarrow O(2^n)$$

Q5
```
int i = 1, s = 1;
while (s <= n)
  { i++, s=s+i;
    printf("#");
  }
```

i = 1 2 3 4 ......

s = 1 3 6 10 ..... n

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

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

$T_K = 1 + 2 + 3 + 4 + \dots K.$

$T_K = \frac{1}{2} K(K+1).$

for K iterations.

$1 + 2 + 3 + \dots + K <= n.$

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

$\frac{K^2 + K}{2} <= n.$

$K^2 <= n$

$K <= \sqrt{n}$

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

Q6

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

as $i^2 <= n$

$\Rightarrow i <= \sqrt{n}$

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

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

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

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

$T(n) = O(n)$

Classmate
Date
Page

Q7

```
void fn (int n)
{  int, i, 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 $K = K * 2$

$k = 1, 2, 4, 8 - - - - n$

G.P $\Rightarrow$ $a = 1$, $r = 2$

Sum of G.P $= \dfrac{a(r^{term} - 1)}{r - 1}$

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

$n = 2^K$

$K = \log n.$

| i | j | K |
|---|---|---|
| $n/2$ | $\log n$ | $\log n * \log n$ |
| $n/2 + 1$ | $\log n$ | $\log n \times \log n$ |
| $n/2 + 2$ | $\log n$ | $\log n \times \log n$ |
| ⋮ | ⋮ | ⋮ |
| $n$ | $\log n$ | $\log n \times \log n.$ |

$\Rightarrow O(\frac{n}{2} * \log n * \log n)$

$\Rightarrow O(n \log^2 n)$, Ans

classmate

Date ______
Page ______

Q8 fn (int n)

```
{ ~~int~~ if (n == 1) return;
      for (i = 1 to n)    // i = 1, 2, 3 ... n
      { for (j = 1 to n)  // j = 1, 2, 3 .... n
          { printf("*");
          }
      }
      function (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

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

for $i = 1 \Rightarrow j = 1, 2, 3, 4 \ldots n = n$

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

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

for $i = n \Rightarrow j = 1 \ldots 1$

$$\Rightarrow \sum_{j=n}^{1} n + \frac{n}{2} + \frac{n}{3} + \frac{n}{4} + \cdots 1$$

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

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

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

$$T(n) = O(n\log n) \text{ // Ans}$$