Design and Analysis of algorithms

(11/03/22)

Ans. 1 Asymptotic notations are used in to find the complexity of an algorithm when input is very large.

- Big O (O): $f(n) = O(g(n))$

  iff

  $f(n) \leq C g(n)$

  $\forall n \geq n_0$

  for some constant $c > 0$

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

- Big omega (Ω): $f(n) = \Omega(g(n))$

  iff

  $f(n) \geq C g(n)$

  $\forall n \geq n_0$

  for some constant $c > 0$

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

- Big Theta (Θ):

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

  iff

  $C_1 g(n) \leq f(n) \leq C_2 g(n)$

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

  for some constant $C_1 > 0$ and $C_2 > 0$.

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

Ans. 2 for ($i = 1$ to n) { $i = i * 2$; }

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

let k^{th} term $= n$

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

taking log on both sides

$\log n = (k-1) \log_2 2$

$k = \log n + 1$

$O(1 + \log n)$

$O(\log n)$ Ans

Ans. 3 $T(n) = 3T(n-1)$ - ①

putting $n = n-1$ in eq. (1)

$T(n-1) = 3T(n-2)$ - ②

put (2) in (1)

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

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

$T(n-2) = 3T(n-3)$ - ③

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

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

$n-K = 0$

$n = K$

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

$= 3^n T(0)$

$= 3^n$

$O(3^n)$ Ans

Ans. 4 $T(n) = 2T(n-1)$ - ①

$n = n-1$ in eq^n (1)

$T(n-1) = 2T(n-2)$ - ②

$T(n) = 4T(n-2)$ - ③

putting, $n = n-2$ in eq. 1

$T(n-2) = 2T(n-3)$ - ④

$T(n) = 8T(n-3)$

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

$n-K = 0$

$K = n$

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

$= 2^n T(0)$

$= 2^n$

$O(2^n)$ Ans

Ans. 5

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

i= 1 2 3 4 5 6 . . .

S= 1 + 3 + 6 + 10 + 15 + . . . + n

sum of s - 1+3+6+10+ . . . +n -(1)

also s = 1+3+6+10 + + . . . +n-1 + n -(2)

from (1) - (2)

0= 1+2+3+4+ . . . n-n

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

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

for K

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

$K(K+1)/2 \leq n$

$(K^2+K)/2 <= n$

$O(K^2) <= n$

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

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

$O(n^{1/2})$ Ans

Ans. 6 void function (int n)
{
int i, count = 0;
for(i=1; i*i ≤ n; i++)
count++;
}

$O(1+\sqrt{n}+\sqrt{n}+\sqrt{n})$

$O(1+3\sqrt{n})$

$O(3\sqrt{n})$

$O(\sqrt{n})$

$O(n^{1/2})$ Ans

Ans. 7 void function (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*1)
count++;
}

i	j	k
n/2	1	1
	↓	2
✓	log n	4
n/2		n
		→ log n

$O(n/2 \times \log n \times \log n)$

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

Ans. 8 function (int n)

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

n	i	j
n	1	j
	⋮	⋮
	2	n
	1	⋮
n-3	n	n
	⋮	⋮
n-6	⋮	⋮

$1+4+7+\dots n$

$n = 1+3(K-1)$

$= 3K-2$

$K = \frac{n+2}{3}$ ✓

no. of terms

$\frac{n+2}{6}\left[2+\frac{(n-1)}{3}\times 3\right]$

$\left[\frac{n+2}{6}(n+1)\right] n^2$

$O\left[\frac{n^2+3n+2}{6} \times n^2\right]$

$O[n^4]$ Ans

Ans. 9

```
void function ( int n )
{
  for (i=1 to n)  n
  {
    for (j=1 ; j <= n ; j = j+1)
        printf("*") ; n2 ;
  }
}
```

$O(n + n^2 + n^2 + n^2)$

$O(3n^2 + n)$

$O(n^2)$ Ans.