Assignment-01

Q: 1

Solution-

Asymptomatic notations are the mathematical notations used to describe the running time of an algorithm when the input tends towards a particular value

There are mainly 3 asymptomatic notations:

Big-O notation.

Omega notation

Theta notation

- Big O(o)

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

  if

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

  $\forall\ n \geq n_0$

  for some constant $c > 0$

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

- Big Omega(Ω):

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

  if

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

  $\forall\ n \geq n_0$

  for some constant $c > 0$.

g(n) is "tight lowest bound" of f(n)

Big Thita Θ

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

if

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

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

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

g(n) is both "high upper & lower bound of f(n)".

Q.2. c

Solution.

values of 'i' in 'for' loop.

1, 2, 4, 27..... 2^k

The loop will terminate when:-

$i \geq n$

$2^k \geq n$

$k \geq \log_2(n)$

so $\log_2(n)$ is total unit of taken by the loop

hence Time complexity = $O(\log_2(n))$

Date
Page

Q.3.

Solution

$$T(n) = 3T(n-1)$$
$$= 3(3T(n-2))$$
$$= 3^2T(n-2)$$
$$= 3^3T(n-3)$$
$$------$$
$$= 3^nT(n-n)$$
$$= 3^nT(0)$$
$$= 3^n$$

$O(3^n)$ Ans.

Q.4.

Solution

$$T(n) = 2T(n-1)-1$$
$$= 2(2T(n-2)-1)-1$$
$$= 2^2(T(n-2)-2-1$$
$$= 2^3T(n-3)-2^2-2^1-2^0$$
$$------$$
$$-------$$
$$= 2^nT(n-n)-2^{n-1}-2^{n-2}-2^{n-3}$$
$$------2^2-2^1-2^0$$

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

Time Complexity is $O(1)$

Q.5.

Solution

Let the following loop execute t times. The loop will executed till i is less than or equal to n.

∴ After first iteration -

$s = s + 1$

After second iteration -

$s = s + 1 + 2$

Since the loop iterates for the t times, we obtain.

$s(t) = 1 + 2 + 3 + \ldots + t <= n$

$= O(t^2) = n$

$= \sqrt{n}$

Hence complexity is $O(\sqrt{n})$

Q.6.

Solution

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

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

$O(3\sqrt{n})$

$O(\sqrt{n})$

Hence complexity $O(\sqrt{n})$

Q.7

Solution

for (i=n/2 ; i<=n ; i++) — will iterate $n/2$ times

for (j=1 ; j<=n ; j*2) → log n times

for (k=1 ; k<=n ; k=k*2) → This loop will execute log n times

count++ ;

$\therefore$ Complexity $= O(n \log^2 n)$

Q.8

Solution

n	i	j
n	1	1
	2	n
(n-3)	n	n
(n-6)		

$1+4+7+ \_\_\_ n$

$n = 1+3(k+1)$

$= 3k-2$

$k = \frac{n+2}{3}.$

No. of terms

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

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

$O\left[n^4\right]$ Ans.

Q. 9.

Solution

```
void function (int n)
{
  for (i = 1 to n) → This loop will execute n times
  {
    for (j=1 ; j<=n ; j=j+1) → This loop will execute n/i
    {
      printf ("*");          for each value of i
    }
  }
```

$$\therefore \text{Complexity} = n \times \sum_{i=1}^{n} \left(\frac{n}{i}\right)$$

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