ASSIGNMENT

Name :- Astha Rai   Section - M1   Page No.   Date   Roll no - 62

Q2:- for(i=1 to n)
{i = i*2}

1 2 4 8 . . . n

$2^{k-1} = n$

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

$k = \log_2 n + 1$

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

Q3:- $T(n) = 3T(n-1)$ $n>0$

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

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

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

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

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

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

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

$n-k=0$

$n=k$

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

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

Q4:- $T(n) = 2T(n-1)$

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

$T(n) = 4T(n-2)$

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

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

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

$n-k=0$

$T(n) = 2^n$ Ans $O(2^n)$

Page No.
Date

Q5:- $i=1$ $s=1$ The value of i increases by 1 at each iteration $n = \frac{k(k+1)}{2} \Rightarrow k = \sqrt{n}$

$O(n)$

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

Q6:-

$i^2 <= n \Rightarrow i <= \sqrt{n}$

$O(\sqrt{n})$ Ans

Q7:- For Ist loop $\rightarrow n/2$

For IInd loop $\rightarrow \log n$

For IIIrd loop $\Rightarrow \log n$

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

Q8:-

n^2 + Recurrence.

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

$T(n-3) = T(n-6)$

$T(n) = T(n-6)$

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

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

$n - 3^k = 0$

~~$n = 1$~~

$n = 3^k$.

$\log n = k \log 3$

$k = \log_n 3$

As $n^2 > \log_n 3$

Time complexity :- $O(n^2)$

Q9:- ~~$O(n^2)$~~

Page No.
Date

$\cancel{1 \cdots\cdots n + n + n/2 + n/3 \cdots\cdots 1}$ 1+

$= n + n/2 + n/3 + n/4 \;.\;.\; n/n$

$= n\left(1 + 1/2 + 1/3 + 1/4 \;.\;.\;.\; + \frac{1}{n}\right)$

$= n(\underbrace{\qquad}_{\log n}) = \underline{n\log n}$

~~Q10~~

Q11:-

```
void fun (int n)
{ int j=1, i=0;
  while (i<n)
  {i=i+j;
   j++; }}
```

i=0 1 3 7

0 1 3 7 12 . .

1

1+2

1+2+4

1+2+4+5

~~n th i~~

last term would be

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

$k^2 = n$

$\boxed{k = \sqrt{n}}$

Time complexity :- $O(\sqrt{n})$ Ans

~~Q12:~~

~~1) nlogn~~

Q14:-

$T(n) = T(n/4) + T(n/2) + cn^2$

~~as T(n/2)~~

Assuming $T(n/2) > T(n/4)$

$T(n) = T(n/2) + cn^2$

Applying master's theorem,

$a=1 \quad b=2$

$f(n) = cn^2 \;|\; n^{\log_1 2}$

As, $cn^2 > n^{\log 2}$

So, $T(n) = O(f(n))$

$T(n) = n^2$ Ans

$= O(n^2)$

Q15:-

$n + n/2 + n/3 + n/4 + \dots$

$n(1 + 1/2 + 1/3 + 1/4 + \dots)$

$= n(\log n) = n\log n$ Ans

Q16:-

$n = i^k$

$\log n = k \log i$

$\boxed{k = \log_i n}$

$O(\log n)$ Ans

Q18:-

a) $100 < \log\log n < \log n < \log(n!) < \text{root}\, n < n < n\log n < n^2 < 2^n < 2^{2n} = 4^n < n!$

b) ~~$\sqrt{\log(n)} <$~~ $1 < \log(\log(n)) < \sqrt{\log(n)} < \log(n) < \log 2n < 2\log n < \log(n!) < n < 2n < 4n < n\log n < n^2 < 2(2^n) < n!$

c) $96 < \log_8 n < \log_2 n < \log(n!) < 5n < n\log_6 n < n\log_2 n < 8n^2 < 7n^3 < 8^{2n} < n!$

Page No.
Date

Q1 Asymptotic notations ~~can be~~ are languages that allow us to analyze an algorithm running time delay identifying is behaviour as the input size of algorithm.

Types :-

1) Big O :- It is commonly used for worst case, and give us upperbond for the growth rate of runtime of algorithm.

Eg :- $O(n^2)$

2) Big Omega :- Provides asymptotic lower bond.

3) Theta :- Tight bond on the growth rate of runtime of algo.

4) Small :- It is used to denote the upper bond. (i.e not asymptotically tight)

$$f(n) = O(g(n)) \; \forall \; f(n) < c\,(g(n)).$$

5) Small Omega :- To denote lower bond

Q2 Ol:

Insertion sort :-

→ Iterative

```
insertion(int arr[], int n)
{
 for(i=1; i<n; i++)
 {
    int val = a[i] - j = i
    while(j>0 && a[j-1] > val)
    { a[j] = a[j-1];
       j--;
 a[j] = val; } }
```

Recursive

```
insertion ( int a [], int i, int n )
{
    int val = a [i], j = i;
    while( j > 0 && a [j - 1] > value )
    {
        a[j] = a [j - 1];
        j--;
    }
    a [j] = val;
    if (i + 1 <= n)
        insertion ( a, i+1, n );
}
```

Q10:- Since polynomials grow slower than exponential n^k has an asymptotic upperbound of $O(a^n)$ for [a = 2, ...]