Page No.
Date : / /20

Design and Analysis of Algorithms

Name : AKSHAT SINGHAL
Class : CST - SPL1
Roll no : 04

Q1

```
void fun (int n)
{
    int j = 1 , i = 0;

    while (i < n) {
        i += j;
        j++;
    }
}
```

Ans

$j = 1$	$i = 1$
$j = 2$	$i = 1 + 2 = 3$
$j = 3$	$i = 3 + 3 = 1 + 2 + 3$
:	
$j = k$	$i = 1 + 2 + 3 + \dots + k$

$\therefore$ as $i < n$,

Sum of k consecutive integers $= \frac{k(k+1)}{2}$

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

$\frac{k^2 + k}{2} < n$ removing constant

$k^2 < n$

$k < \sqrt{n}$

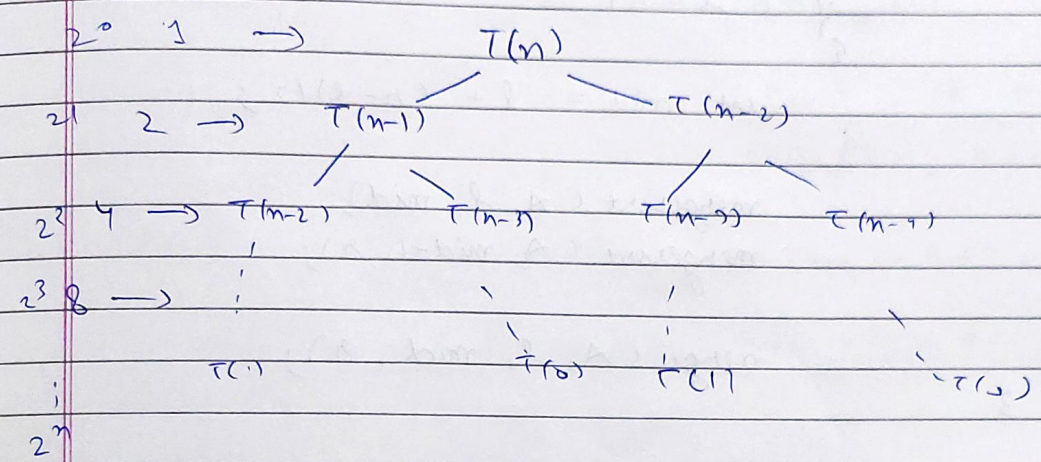
$T(n) = O(\sqrt{n})$

Page No.
Date : / /20

Q2. Write Recurrence Relation for the recursive function that prints Fibonacci series. Solve the recurrence relation to give time complexity of the program. What will be the space complexity of this program & why?

Sol Recurrence Relation:

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



$$T(n) = 2^0 + 2^1 + 2^2 + 2^3 \cdots 2^n$$

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

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

$$\boxed{T(n) = O(2^n)}$$ ✓

Above space complexity depends on the maximum depth of the tree

$$S.C = O(n)$$ ✓

Q3. Write programs which have complexity $n(\log n)$, n^3, $\log(\log n)$.

Soln. is $n(\log(n))$

```
void mergesort (int A[ ] int l, int r)
{
    if (l < r)
    {
        int mid = l + (r-l)/2;

        mergesort (A, l, mid);
        mergesort (A, mid+1, r);

        merge (A, l, mid, r);
    }

void merge (int A[ ], int l, int mid, int r)
{
    int c[r];
    int i = l, j = m+1, k = 0;

    while (i <= mid && j <= r)
    { if (a[i] < a[j])
        c[k++] = a[i++];
      else
        c[k++] = a[j++];
    }
while (i <= mid)
    c[k++] = a[i++];
while (j <= r)
    c[k++] = a[j++];
for (int i = 0; i < k; i++)
{   a[l+i] = c[i];
}
```

ii) n^3

```
void pattern (int n)
{
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
        {
            cout << "#" << " ";
            for (int k=0; k<n; k++)
            {
                cout << "*" << " ";
            }
            cout << "#" << " ";
        }
        cout << endl;
    }
```

iii) log(log n).

```
for (int i=0; i<n; i=i*2)
{
    for (int j=n; j>1; j=j/2)
    {
        // O(1)
    }
}
```

4. Solve the following Recurrence Relation $T(n) = T(n/4) + T(n/2) + cn^2$

$T(n/2) \simeq T(n/4)$

Neglecting $T(\frac{n}{4})$ term as lower order

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

Now, using Master's theorem

$T(n) = aT(\frac{n}{b}) + f(n)$

$a = 1$

$b = 2$

$c = \log_b a = \log_2 1 = 0$

$c = 0$, $n^c = n^0 = 1$

$f(n) = cn^2$

$f(n) > 1$

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

6. What should be time complexity of

```
for (int i=2 ; i<=n; i = pow(i, k))
{
    // some O(1)
}
```

where, K is a constant.

i) $i = 2$

ii) $i = 2^k$

iii) $i = (2^k)^k = 2^{k^2}$

iv) $i = (2^{k^2})^k = 2^{k^3}$

:

$i = 2^{k^n}$

$2, 2^k, 2^{k^2} \; - \; - \; - \; 2^{k^{n-1}}$

$\rightarrow 2^{k^{n-1}} <= n$

$\rightarrow k^{n-1} = \log_2 n$

$t = \log_k \log_2 n \rightarrow T(n) = O(\log_k(\log_2(n)))$ Ans

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

$\log\log n < \log n < \log^2(n) < \sqrt{n} < 100 < n < \log(n!) < n\log n < n^2 < 2^n < n! < 4^n = 2^{2n}$

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

$\log(\log(n)) < 1 < \sqrt{\log(n)} < \log n < \log 2n < 2\log n < n < \log(n!) < n\log n < n^2 < 2.2^n < n!$

$2^n < 4^n$

Page No.
Date : / /

Q. 8^{2n}, $\log_2(n)$, $n \log_8(n)$, $n \log_2(n)$, $\log(n!)$, $n!$, $\log_8(n)$, 96, $8n^2$, $7n^3$, $5n$

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

5. What is the time complexity of following function fun

```
int fun (int n)
{
    for (int i = 1; i <= n; i++)
    {
        for (int j = 1; j < n; j += i)
        {
            // Some O(1) task
        }
    }
}
```

$i = 1$, $j = \underbrace{1, 2, 3, \ldots, n}_{n \text{ times}}$

$i = 2$, $j = \underbrace{1, 3, 5, \ldots, n}_{\frac{n+1}{2}}$

$n = 1 + (t-1) \times 2$

$t = \frac{n+1}{2}$

$i = 3$, $j = \underbrace{1, 4, 7, \ldots, n}_{\frac{n+2}{3}}$

$i = n$, $j = \underbrace{1}_{\frac{n+n-1}{n}} = 1$

Page No.
Date : / /20

$$T.C = n + \frac{n+1}{2} + \frac{n+2}{3} + \frac{n+3}{4} \quad \cdots \quad \frac{n+k-1}{k}$$

$$T.C = \sum_{i=0}^{i=n} \frac{n+k-1}{k}$$

$$T.C = \sum_{i=0}^{i=n} \frac{n}{k} + \sum_{i=0}^{i=n} (1) + \sum_{i=0}^{i=n} \left(\frac{1}{k}\right)$$

$$T.C = n \log n + n - \log n$$

Now neglecting lower order terms

$$T.C = o(n \log n)$$

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

Ans