Assignment-2

Name - Vibha Rani
Rollno - 45
Section - CE [Computer Engineering]
University Rollno - 2017395

Tutorial-2

Ques1 → find time complexity

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

$j=1$	$i=1$
$j=2$	$i=1+2=3$
$j=3$	$i=3+3=6$
$j=4$	$i=6+4=10$
	⋮
$j=k$	$i=(k-1)+k$

$s = 1+3+6+10+ \cdots (k-1)+k$

$-s = 1+3+6+ \cdots +(k-1)+k$

$0 = 1+2+3+4+ \cdots n-k$

$k = 1+2+3+4+ \cdots k$

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

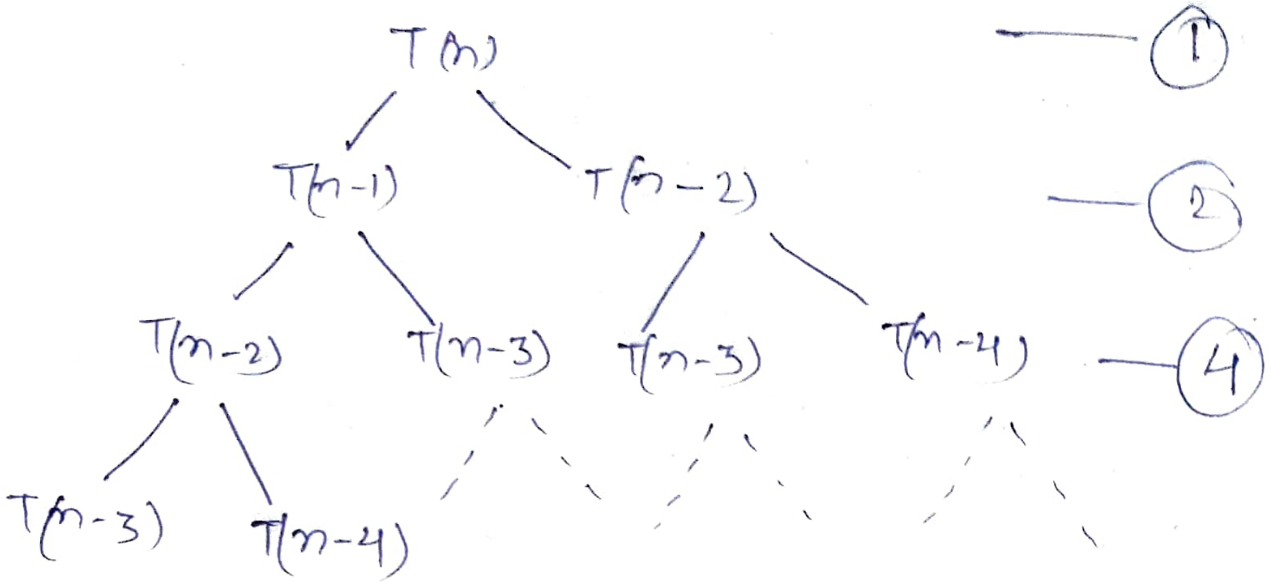
$\frac{k^2+k}{2} < n$

$k \leq \sqrt{n2}$

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

Ques 2 Recursive relation for fibonacci series:

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



$\Rightarrow 1 + 2 + 4 + 8 + \cdots$

$a = 1 \quad r = 2$

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

$T(n) = \cancel{\Theta 2n} \; O(2^n)$

Ques 3: Write programs having following complexities

(i) n(logn)

```
void quick-sort(int a[], int lb, int ub)
{
     int i= lb, j=ub;
      int key = a[lb];
       int t=0;
       if (lb>=ub)
            return;
      while(i<j)
     {    while(key>= a[i] && i<j)
              i++;
```

```
    while (key < a[j])
        j--;
    if (i<j)
    {   t = a[i];
        a[i] = a[j];
         a[j] = t;
    }
  }
  a[lb] = a[j];
  a(j) = key;
  quick-sort (a, 0, j-1);
  quick_sort (0, j+1, ub);
}
```

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

```
for ( int i=0; i<n; i++)
{
    for ( int j =0; j<n; j++)
    {
        for ( int k=0; k<n; k++)
        {
            sum += k;
        }
    }
}
```

(iii) $O\log(\log n)$

```
int p=0;
for ( int i=1; i<n; i= i*2)
{
    p++;
}
for ( int j=1; j<p; j= j*2)
{
        //O(1) operation.
}
```

Ques 4- $T(n) = T(n/4) + T(n/2) + n^2$

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

Using Master's method

$T(n) = aT(n/b) + f(n)$

$a=2$

$b=2$

$c = \log_b a = 1$

$$\boxed{T(n) = f(n) = O(n^2)}$$

Ques 5 $i = 1\ 2\ 3\ 4\ \dots\ n$

$j = 1, 2, 3, \dots$ n times

$T(n) = n + n/2 + \frac{n}{3} + \frac{n}{4} + \dots + 1$

$n\left[1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n}\right]$

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

Ques 6 $T(n) = 2, 2^{k}, 2^{k^2}, 2^{k^3} \ldots 2^{k \log_k(\log n)}$

So $2^{k \log_k \log n} = 2^{\log n} = n$

So Total Time complexity

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

Ques 7

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

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

(iii) $96 < \log_2(n) < \log_2 n < 5n < n \log_2 n < n \log_2 n <$
$n! < \log n! < 8^{2n}$.