Himanshi Rana
ROLL NO - 20
SPL - 2

TUTORIAL - 2

Q1)
```
void fun (int n)
{ int j=1, i=0;
  while(i<n)
  { i = i+j;
    j++;
  }
}
```

Values after execution —

1st time → $i = 1$

2nd time → $i = 1+2$

3rd time → $i = 1+2+3$

4th time → $i = 1+2+3+4$

ith time → $i = (1+2+3+4 \ldots i) < n$

$$\Rightarrow \frac{i(i+1)}{2} < n$$

$$\Rightarrow \frac{i^2+i}{2} < n$$

$$i^2 < n$$

$$i = \sqrt{n}$$

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

(Q 2)

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

Let time taken =

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

$T(n-1) \approx T(n-2)$

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

$T(0) = T(1) = 0$

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

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

$T(n-2) = 2T(n-3) + 1 \rightarrow (3)$

Put (2) in (1)

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

$= 4T(n-2) + 3 \rightarrow (4)$

Put (3) in (4)

$T(n) = 4(2T(n-3)+1) + 3$

$= 8T(n-3) + 7$

General eq.

$2^k T(n-k) + (2^k - 1)$

for $T(0)$

$n-k=0 \Rightarrow k=n$

$T(n) = 2^n \times T(0) + 2^n - 1$

$= 2^n - 1 = O(2^n)$ Ans

Space complexity = $O(N)$

Q3 (i) n(logn)

```
int sum =0, n =8;
for( int i =1; i<=n; i++)
 {
 for (int j =1; j<=n ; j* =2)
   {
     sum+ =j;
   }
 }
```

(ii) n 3

```
int sum = 0, n= 8;
for (int i = 1 ; i<=n; i++)
 {
 for ( int j = 1 ;j <= n; j++)
  {
  for (int K = 0; j <=n; k++)
    {
      sum++;
    }
  }
 }
```

(iii) log (log n)

```
int sum = 0, n= 8;
for (int i =1 ;i <=n ; i* =2)
 {
 for (int j =1 ;j <=n ; j * =2)
   {
     sum+ =j;
   }
 }
```

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

Using Master's Theorem,

We can assume $T(n/2) >= T(n/4)$

Equation can be written as

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

$\Rightarrow T(n) <= O(n^2)$

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

Also $T(n) >= cn^2 \Rightarrow T(n) >= O(n^2)$

$\Rightarrow T(n) = \Omega(n^2)$

$\therefore T(n) = O(n^2)$ and $T(n) = \Omega(n^2)$

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

Q5) for $i = 1$, inner loop is executed n times

for $i = 2$, inner loop is executed $n/2$ times

for $i = 3$, inner loop is executed $n/3$ times.

$\Rightarrow n + \frac{n}{2} + \frac{n}{3} + \cdots + \frac{n}{n}$

$n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots \frac{1}{n}\right)$

$\Rightarrow n \times \sum_{k=1}^{n} \frac{1}{k}$

$\Rightarrow n \times \log n.$

Q6 for (int i = 2; i <= n; i = pow(i, k))

{

 // same as O(1) expressions

}

with iterations

i takes values

for 1 iteration → 2

for 2 iteration → 2^k

for 3 iteration → $(2^k)^k$

for n iterations → $2^{k \log_k(\log n)}$

Last term must be less than or equal to n

$2^{k \log_k(\log n)}$

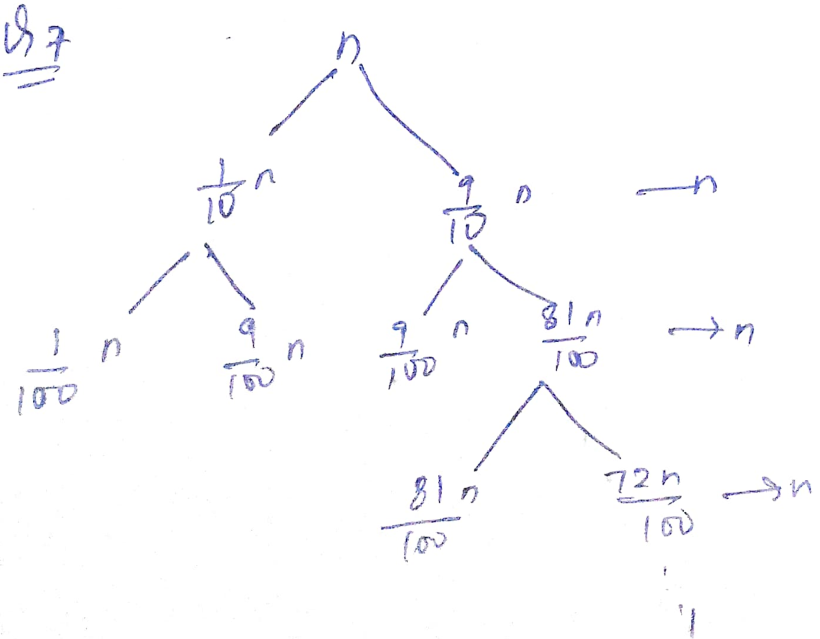
$= 2^{\log n} = n$

Each iteration takes constant time

∴ Total iteration = $\log_k(\log(n))$

Time complexity = $O(\log(\log(n)))$ Ans

Q7



Recurrence Relation

$T(n) = T(9n/10) + T(n/10) + O(n)$

Where first branch is of size $\frac{9n}{10}$ and second one is $\frac{n}{10}$

Solving the above using ~~[illegible]~~ recursion tree.

At 1st level, value $= n$

At 2nd level, value $= \frac{9n}{10} + \frac{n}{10} = n$

Value remains same at all levels i.e. n

Time complexity = summation of values

$= O(n \times \log_{10/9} n)$ (upper bound)

$= \Omega(n \log_{10} n)$ (lower bound)

$\Rightarrow O(n \log n)$ Ans

Q8 Considering large values of n

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

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

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