NAME:- AYUSH RAJ

UNIVERSITY ROLL NO.:- 2017317

SECTION:- CE   CLASS ROLL NO.:- 20

ASSIGNMENT:- 02

Ans 01)

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

$j=01$; $i=0+1$

$j=2$; $j=0+1+2$

$j=3$; $j=0+1+2+3$

⋮

$j=n$; $\boxed{1 \times = n}$ HW

$0+1+2+\cdots n > n$

$\frac{K(K+1)}{2} > n$

$K^2 > n$ $K > \sqrt{n}$

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

Ans 02) Recurrence rel^n for Fibonacci series:-

$T(n) = T(n-1) + T(n-2)$ $T(0) = T(1) = 1$

Assume $T(n-1) \simeq T(n-2)$

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

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

$= 4[2T(n-6)] = 8T(n-6)$

⋮

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

$n-2K=0$

$n = 2K$

$K = n/2$

$T(n) = 2^{n/2} T(0)$

$T(n) = 2^{n/2}$

$T(n) = \Omega(2^{n/2}) = 2^{n/2}$

if $T(n-2) \simeq T(n-1)$

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

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

⋮

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

$n-1 \le 0$

$\boxed{K=n}$

$T(n) = 2^K T(0)$

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

$\boxed{O(n) = 2^n}$ Ans.

Ans 03)

```
for (i=0; i<n; i++)
{
for (j=1; j<n; j=j*2)
{
  //some O(1)
}
}
```
— $O(n\log n)$.

```
for (i=0; i<n; i++)
{
for (j=0; j<n; j++)
{
for (k=0; k<n; k++)
{
//some O(1);
}
}
}
```
— $O(n^3)$.

```
for (i=1; i<=n; i=i*2)
{
for (j=1; j<=n; j=j*2)
{
  //some O(1);
}
}
```
— $O(\log(\log n))$.

Ans 04) $T(n) = T(n/4) + T(n/2) + Cn^2$

lets assume $T(n/2) >= T(n/4)$

so; $T = 2T(n/2) + Cn^2$

Applying Master's theorm;

$a = 2$ $\quad f(n) = n^2$
$b = 2$

$c = \log_b a = 1.$

$n^c = n$

Comparing;

$f(n) > n^1$

also, $\boxed{T(n) = \Theta(n^2).}$ — Ans

Ans 05.)

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

$i=1$	2	3	4	
$j=1$	1	1	1	
⋮	3	4	⋮	
n	5	7	$n/4$	— n^2
	7	⋮		
	⋮	$n/3$		
	$n/2$			

Hence; $T(n) = O(n^2) + O(n^2/2) + O(n^2/3) + \dots$

$\boxed{T(n) = O(n^2)}$ — Ans.

Ans 06.)

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

$pow(i,K) = O(\log n)$
$= \log k$

Loop ends $2^{k^m} > n$

$\log 2^{k^m} > \log n$ — taking log both sides.

$K^m \log 2 > \log n$

$\log K^m > \log(\log n)$

$m \log K > \log(\log n)$

$m > \frac{\log(\log n)}{\log K}$

Hence;

$\boxed{T(n) = \log_2(\log_2 n)}$ — Ans

Ans 07.) Considering the statement;

$T(n) = T(n/100) + T(99n/100) + O(n)$

where $n/100$ & $99n/100$ are for parts & $O(n)$ is partitioning algo.

n ——— 1st step – n

$\frac{n}{100}$ $\quad$ $\frac{99n}{100}$ ——— 2nd step – n

$\frac{(n/100)}{100}$ $\quad$ $\frac{99(n/100)}{100}$ $\quad$ $\frac{(99n/100)}{100}$ $\quad$ $\frac{99(99n/100)}{100}$ ——— 3rd step – n

So, it will remain n at each step.

So; time complexity $= O(n * \log_{100/99} n)$ if we take longer branch

$= \Omega(n * \log_{10} n)$ — TIME COMPLEXITY.

Question 08>

a> Order is:-

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

(b) Order is.-

$1 < \sqrt{\log n} < \log n < 2\log n < \log_2 N < N < 2N < 4N < \log(\log N) < N\log N < \log N! < N! < N^2 < 2 \times 2^N.$

(C) Order is.-

$96 < \log_8 N < \log_2 N < n\log_6 N < n\log_2 N < \log n! < N! < 5N < 8N^2 < 7N^3 < 8^{2n}.$