Subject : Sheet (1) Solution Date: / /

Question 1

[A] For (int i=0; i<n, i++)
cout << i;

instruction	cost	No. of operations
int i=0	1	1
i<n	1	n+1
cout << i	1	n
i++	1	n

Total No. of operations = 3n+2

Running time → O(n)

[B] For (int i=1 ; i<n ; i *= 2)
cout << i;

instruction	cost	No. of operations
int i=1	1	1
i<n	1	$\log_2 n + 1$ → stop cond. check
cout<<i	1	$\log_2 n$
i *= 2	1	$\log_2 n$

let's observe i

i
1
$1\times2 = 2$
$2\times2 = 2^2$
$2^2\times2 = 2^3$
2^k

assume $i >= n$

$\therefore 2^k >= n$

loop stopping cond. $2^k = n$

$\therefore k = \lceil \log_2 n \rceil$ → ceil

Running time → $O(\log_2 n)$

→ Why $k = \lceil \log_2 n \rceil$ (ceil Value)

assume n=8		n=10	
Values of i	1, 2, 4 (3)	1, 2, 4, 8	(4)
stop ← 8		16 → stop	
$\boxed{k=3}$		$\boxed{k=4}$	

But $\log_2 10 = 3.32$

Subject : .... Date: / /

C

```
for (int i=0; i<n; i++)  -> n+1
{
    F(n);  -> log n * n
}
```

Note: F(n) has O(log n)

Total = $n\log n + \cancel{n} + \cancel{1}$

Running time $\rightarrow O(n\log n)$

D

```
void decimal2binary(int n)  -> T(n)
{
    if(n>0)
    {
        decimal2binary(n/2);  -> T(n/2)
        cout << n%2;          -> 1
    }
}
```

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

$$T(n) = \begin{cases} 1 & n=0 \\ T(n/2)+1 & n>0 \end{cases} \rightarrow \text{Recurrence Relation}$$

* By Applying Master Thearom (Dividing Function)

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

→ By Comparing it to Master Thearom then $a=1$ $k=0$ $b=2$ $p=0$

$\therefore \log_b a = k = 0 \xrightarrow{\text{case}} O(n^k \log^{p+1} n)$

$O(n^0 \log^1 n) = O(\log n)$