Tutorial-2 (DAA)

Prapti Goel
Sec-I
Roll No.-66

Answer 1) → Void function (int n)

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

i = 1, i = 0+1
i = 2, i = 0+1+2
i = 3, i = 0+1+2+3

Loop ends when $i \geq n$

$0+1+2+3 \cdots n > n$

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

$K^2 > n$

$K = \sqrt{n}$

$O(\sqrt{n})$.

Answer 2) →

Recurrence Relation For Fibonacci Series:

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

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

⊙ If $T(n-1) \approx T(n-2)$

(Lower Bound)

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

$= 2 \langle 2T(n-4) \rangle = 4T(n-4)$

$= 4(2T(n-6))$

$= 8T(n-6)$

$= 8(2T(n-8))$

$= 16T(n-8)$

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

$n - 2^K = 0$

$n = 2K$

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

$= 2^{n/2}$

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

⊙ if $T(n-2) \approx T(n-1)$

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

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

$$= 4(2T(n-3)) = 8T(n-3)$$

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

$$n - k = 0$$

$$\boxed{k = n}$$

$$T(n) = 2^k \times T(0) = 2^n$$

$$= T(n) = O(2^n) \quad \text{(upper Bound)}$$

Answer 3) → ⊙ $O(n(\log n))$ => for (int i = 0; i < n; i + 1)

{

for (int j = 1; j < n; j = j * 2)

{

// some O(1)

}

}

⊙ $O(n^3)$ => for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; i++)

{

for (int k = 0; k < n; k++)

{

// some θ(1)

}

}

}

⊙ $O(\log(\log n))$ => for (int i = 1; i <= n; i = i * 2)

{

for (int j = 1; j <= n; j = j * 2)

{

// some O(1)

}

}

Answer 4) → $T(n) = T(n/4) + T(n/2) + cn^2$.

Let's assume $T(n/2) >= T(n/4)$

So, $T(n) = 2T(n/2) + cn^2$

Applying Master's Theorem $\left(T(n) = aT\left(\frac{n}{b}\right) + f(n)\right)$

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

$c = \log_b^a = \log_2^2 = 1$

$n^c = n$

compare n^c and $f(n) = n^2$

$f(n) > n^c$

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

Answer 5)

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

$i = 1$ — $i = 1, i = 2, i = 3, \ldots, i = n$ — n times

$i = 2$ — $j = 1, j = 3, j = 5, j = 7$ — Loop ends when $j > n$

$1 + 3 + 5 + 7 > n$

$K > \frac{n}{2}$

— n times

So, Total Complexity =

$= O(n^2 + n^2 + n^2 \ldots)$

$= O(n^2)$

$i = 3$ — $j = 1, j = 4, j = 7$ — $1 + 4 + 7 > n$, $K > \frac{n}{3}$

$i = 4$ — $K > \frac{n}{4}$

$i = n$.

Answer6) → for (int i=2; i<=n; i= pow(i,K))

& // same (1)

&

complexity of pow(i,K) — O(log N)
= log(K)

$i = 2$

$i = 2^K$

$i = 2K^2$

$i = 2K^3$

$i = 2K^4$

$i = 2K^n$

$i = 2K^M$

loops ends when $i > n$

$2^{K^M} > n$

$\log(2^{K^n}) > \log n$

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

$K^M > \log n$

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

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

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

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

Answer8) → a) $100 < \log n < \sqrt{n} < n < \log(\log n) < n \log n$
$< \log n! < n! < n^2 < \log^{2n} < 2^n < 2^{2n} < 4^n$.

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

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