DAA Tutorial Sheet - 1

Name- Swapnil Srivastava
Course- B.Tech CSE
Roll No.- 1961177
Section - B

A1 → (3) $O(N+M)$ time
$O(1)$ space

A2 → $T(n) = O(n)$, space $O(1)$

A3 → $T(n) = O(\log_2 n)$, space $O(1)$

A4 →

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

$= n + (n-1) + (n-4) + (n-9) + \cdots (n-k)$

$= n + (n*k) - (1^2 + 2^2 + 3^2 + \cdots k^2)$

$= \sqrt{n}$

$i^2 < n$

$i < \sqrt{n}$

$T(n) = O(\sqrt{n})$, space $O(1)$

A5 →

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

0<=n   1

1<=n   1

3<=n

$$\underbrace{(0, 1, 3, 6, 10, 15, 21, \ldots n)}_{k\text{-term}}$$

$$k^{th}\ term = \left(\frac{k*(k+1)}{2}\right)$$

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

$$k^2+k = 2n$$

$$k^2+k-2n = 0$$

$$k = \frac{\pm\sqrt{1^2+8n}}{2}$$

$$k = \frac{\sqrt{8n+1} \quad 1}{2}$$

$$k = \frac{\sqrt{8n+1}}{2}$$

$$k = \frac{\sqrt{8n}}{2} = \sqrt{n}$$

$T(n) = \sqrt{n}$ space - O(1)

A6 → void Recursion (int n) → T(n)

```
{
   if (n==1) return;
   recursion (n-1)       → T(n-1)
   print (n);            → 1
   recursion (n-1);      → T(n-1)
}
```

$$T(n) = \begin{cases} 1 & n = 1 \\ 2T(n-1)+1 & n > 1 \end{cases}$$

$T(n) = 2T(n-1)+1$ — ①

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

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

$T(n) = 4T(n-2)+(1+2)$ — ②

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

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

$T(n) = 8T(n-3)+(1+2+4)$ — ③

$T(n) = 8[2(T(n-4)+1)+[1+2+4]$

$T(n) = 16T(n-4)+[1+2+4+8]$ — ④

$T(n) = 2^k T(n-k) + (\underbrace{1+2+4+8+ \ldots}_{k\text{-times}})$

Scanned by TapScanner

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

$T(n) = 2^{n-1} T(1) + (1+2+4+8 \cdots)$
(n-1) times.

$T(n) = \frac{2^n}{2} + (1+2+4+8 \cdots)$
(n-1) times.

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

$T(n) = \frac{2^n}{2} + \left(\frac{2^{n-1} - 1}{1}\right)$

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

$T(n) = \cancel{2}\left(\frac{2^n}{\cancel{2}}\right) - 1$

$T(n) = 2^{n-1} + \left(1 \left(\frac{2^{n-1} - 1}{1}\right)\right)$

$T(n) = 2(2^{n-1}) - 1$

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

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

A7 → It is a binary search algorithm.

$T(n) = \log_2 n$

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

by using Masters Method (can't be solved)

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$.

So $a = 1$
$b = 2$
$f(n) = 1$

$c = \log_b a = \log_2 1 = 0$

$0 \simeq 1$

$n^0 = f(n) = 1$

$\therefore \; n^c = f(n)$

$T(n) = O(\log_2 n)$

A8 → $T(1) = 1$

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

$n - k = 1$
$k = n - 1$

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

$T(n) = n$
$T(n) = O(n)$

(2) $T(n) = T(n-1) + n$ — (1)

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

$T(n) = T(n-2) + (n + (n-1))$ — (2)

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

$T(n) = T(n-k) + (n + (n-1) + (n-2) + \cdots (n-k-1))$

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

$n = k+1$

$k = n-1$

$T(n) = T(1) + (n + (n-1) + (n-2) \cdots (\cancel{n} - \cancel{n} + \cancel{1} - \cancel{1}))$

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

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

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

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

A8 →

Ans3 → $T(n) = T(n/2) + 1$ — (1)

$T(n/2) = T(n/4) + 1$

$T(n) = T(n/4) + 2$ — (2)

$T(n/4) = T(n/8) + 1$

$T(n) = T(n/8) + 3$ — (3)

$T(n) = T\left(\frac{n}{2^k}\right) + k$ — (4)

$\frac{n}{2^k} = 1$

$2^k = n$

$k = \log_2 n$

$T(n) = T(1) + \log_2 n$

$T(n) = O(\log_2 n)$

<u>A8</u> →

(Ans4) → $T(n) = 2T\left(\frac{n}{2}\right) + 1$

$c = 1$

$n^c = n$

$f(n) = 1$

$n^c > f(n)$

$T(n) = \Theta(n)$

<u>A8</u>

(Ans 5) → $T(n) = 2T(n-1) + 1$

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

Scanned by TapScanner

A8 →

(Ans 6) → $T(n) = 3T(n-1)$, $T(0) = 1$

$T(n) = 3T(n-1)$ — (1)

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

$T(n) = 9T(n-2)$

$T(n) = 3^3 T(n-3)$

$T(n) = 3^k T(n-k)$

for $n-k=0$

$n=k$

$T(n) = 3^n T(0)$

$T(n) = 3^n$

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

A8 →

(Ans 7) →

$$T(n) = \begin{cases} 1 & , n <= 2 \\ T(n) & n > 2 \end{cases}$$

$T(n) = T(\sqrt{n}) + 1$ — (1)

$T(\sqrt{n}) = T(n^{1/4}) + 1$

$T(n) = T(n^{1/4}) + 2$ — (2)

$T(n) = T(n^{1/8}) + 3$ — (3)

$T(n) = T(n^{(1/2)}) + k$

for $T((\sqrt{n})^{1/k}) = T(2)$

$n^{1/2k} = 2$

$n^{1/2k} = 2$

$\frac{1}{2k} \log n = 1$

$2k = \log n$

$k = \log_2 (\log n))$

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

A8 →

Ans 8 → $T(n) = T(\sqrt{n}) + n$

$T(\sqrt{n}) = T(n^{1/4}) + \sqrt{n}$

$T(n) = T(n^{1/4}) + (n + \sqrt{n})$

$T(n) = T(n^{1/8}) + (n + \sqrt{n} + n^{1/4})$

$T(n) = T(n^{1/2k}) + \underbrace{(n + n^{1/2} + n^{1/4} + \dots)}_{k \text{ terms}}$

for $n^{1/2k} = 2$

$\frac{1}{2k} = 1/\log(n)$

$2k = \log(n)$

$k = \log(\log(n))$.

$T(n) = 1 + (n + \sqrt{n} + \sqrt{n}\sqrt{n} + \cdots)$ k terms.

$T(n) = 1 + \left(\text{GP } \begin{array}{l} a = n \\ r = \sqrt{n} \\ \text{No. of terms} = k \end{array} \right)$

$T(n) = 1 + \left(\frac{n((\sqrt{n})^k - 1)}{(k-1)} \right).$

$T(n) = 1 + n\left(\frac{(\sqrt{n})^{\log\log(n)} - 1}{\log\log(n) - 1} \right)$

$T(n) = n \cdot \log\log(n)$ {by neglecting other values}.

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

A9 →

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

0, 1, 2, ... n

so $T(n) = O(n)$, space $O(1)$.

A10 → $O(N * (N, N-1, \ldots, 1))$

$O(N * (\frac{N+1}{2}))$

(4.) $O(N*N)$

Scanned by TapScanner

A11 → $O(\frac{n}{2} * (\log 2N))$

$O(n \log n)$

A12 → (2) X will always be a better choice for large input.

A13 → (4) $O(\log N)$

A14 → $T(n) = 7(T(\frac{n}{2})) + (3n^2+2)$

$f(n) = 3n^2+2$

$a=7$

$b=2$

$c = \log_b a = \log_2 7 = 2.807$.

$n^c = n^{2.8} \approx n^{2.8}$

$f(n) = 3n^2+2$

so $n^c > f(n)$

so $T(n) = \Theta(n^{2.8})$ or (c) $\Theta(n^{2.8})$

(a) $\Theta(n^{2.8})$

(d) $\Theta(n^3)$

A15 → $f1(n) = n^{\sqrt{n}}$

$f2(n) = 2^n$

$f3(n) = (1.000001)^n$

$f4(n) = n^{(10 * 2^{n/2})}$.

(a) $f_2(n) > f_4(n) > f_3(n) > f_1(n)$.

A16 → $f(n) = 2^{2n}$

$\log f(n) = 2n \log_2 2$

$\log f(n) = 2n$

or

$f(n) = 2^n \cdot 2^n$

$\Omega(2^n)$.

A17 → $T(n) = 2T\left(\frac{n}{2}\right) + n^2$

$c = 1$

$n^c = n$

$n^2 > n$

$f(n) > n^c$

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

A18 → $O(\log N)$ [It's a G.C.D. function where n keeps on decreasing by $n/2$].

A19 → $T(n) = O(N^2 + N)$

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

Scanned by TapScanner