11) Recurrence relation $T(n) = T(n-1) + n$

Decreasing func.

```
void Test(int n)                    ——  T(n) time.
{
  if(n>0)                           ——  1
  {
    for(i=0; i<n; i++)              ——  n+1
    {
      printf("%d", n);              ——  n
    }
    Test(n-1);                      ——  T(n-1)
  }
}
```

∴ Recurrence relation –

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

In order to simplify, since $2n+2$ belongs to linear class, we can write only n.

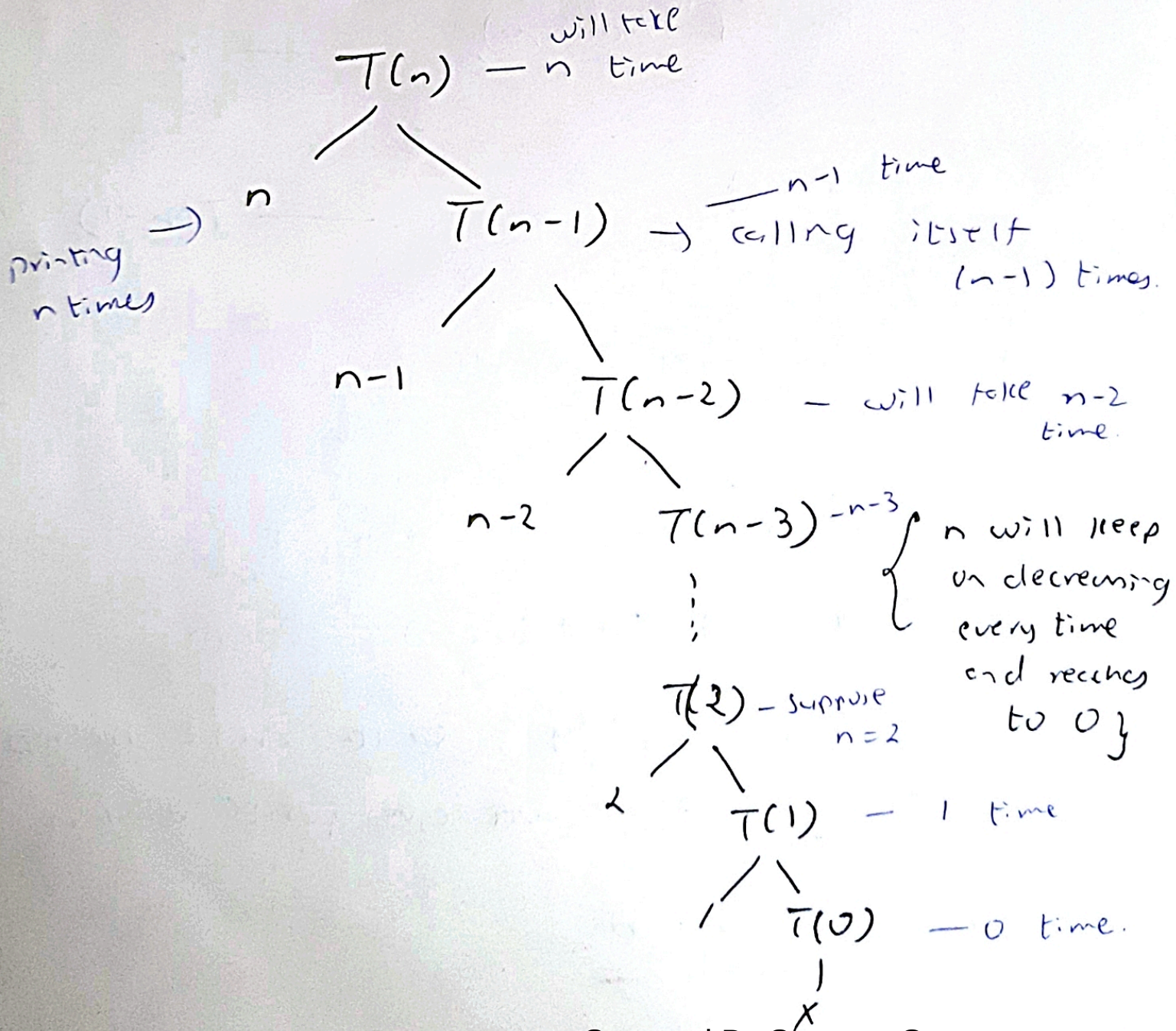
∴ $T(n) = T(n-1) + n$

It can be written as $O(n^2)$

$$T(n) = \begin{cases} 1 & n = 0 \\ T(n-1) + n & n > 0 \end{cases} \quad -(1)$$ // Recurrence relation

So we have prepared recurrence relation for a given program or algorithm.

→ Now solving (1) using Recursion Tree



Summing all

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

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

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

$$= \Theta(n^2) \text{ or } O(n^2)$$

Solved using Recurrence tree

Solving above by substitution method –

$$T(n) = \begin{cases} 1 & n = 0 \\ T(n-1)+n & n > 0 \end{cases}$$

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

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

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

(In place of n, put n-1)

→ So substitute $T(n-1)$ to $T(n-2)+n-1$

$$\therefore T(n) = [T(n-2)+n-1] + n \quad -(1)$$

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

Substitute for this also.

$$\therefore T(n-2) = T(n-3) + n-2$$

After substituting

$$\therefore T(n) = [T(n-3)+n-2] + (n-1) + n$$

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

So after substitution for 2 times, we have an idea of sequence. So if we do it for k times then -

⋮ k times

→ $T(n) = T(n-k) + (n-(k-1)) + (n-(k-2)) + \dots + (\text{[illegible]}) + n. \quad -(4)$

Similar to $n-2$. as $n-2$ is 1 less than $n-3$. so it is $k-1$.

→ Assume $n-k=0$

$\therefore n=k.$

→ In (4) substitute $n=k$.

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

$\therefore T(n) = T(0) + 1+2+3+\cdots+n$

sum of n nos.

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

$= O(n^2)$

Same as recursive tree method.

III)

Consider an example —

$T(n) = T(n-1) + \log n.$

- Decreasing func.

→
```
void Test(int n)          — will take T(n) time
{
  if(n>0)
  {
    for(i=1; i<n; i=i*2)
    {
      printf("%d", i);    — log n
    }
    Test(n-1);            — T(n-1)
  }
}
```

→ $T(n) = T(n-1) + \log n$

∴ Recurrence relation is –

$$T(n) = \begin{cases} 1 & n=0 \\ T(n-1)+\log n & n>0 \end{cases}$$

→ Solving using Tree method –