Topics to discuss

① Divide and Conquer

② Recurrence Relation

③ How to write recurrence relation.

④ Methods to solve Recurrence Relation.

Divide-and-Conquer

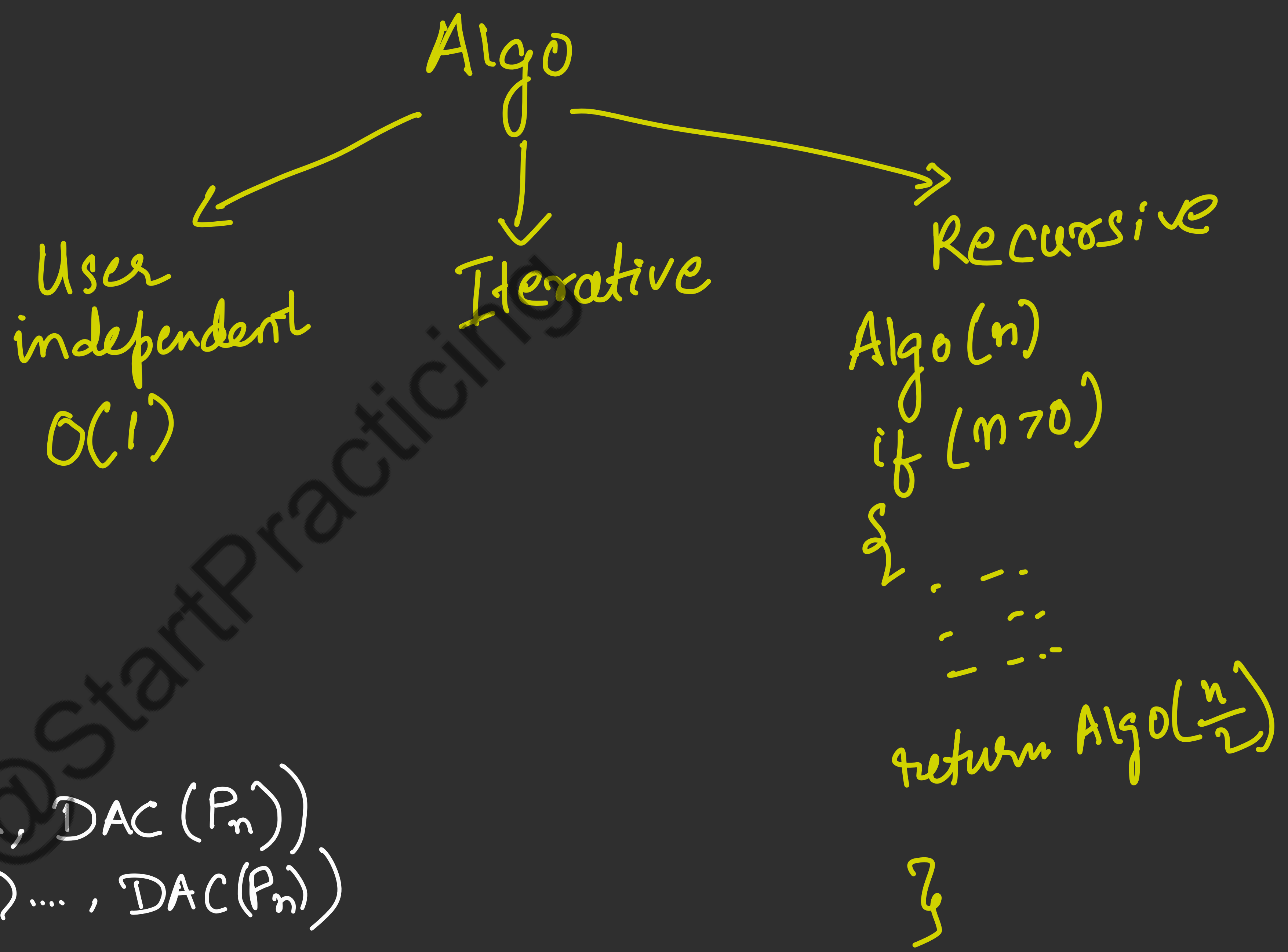
Divide : The problem into a number of subproblems that are smaller instances of the same problem.

Conquer : The subproblems by solving them recursively. If the subproblem sizes are small enough, however, just solve the subproblems in a straightforward manner.

Combine : The solutions to the subproblems into the solution for the original problem.

General Form :

```
DAC (P)
{
    if (small (P))
    {
        solve (P)
    }
    Else
    {
        divide P into $P_1, P_2, P_3 \dots P_n$
        Apply (DAC ($P_1$), DAC ($P_2$), ...., DAC ($P_n$))
        Combine (DAC ($P_1$), DAC ($P_2$) .... , DAC ($P_n$))
    }
}
```



```
Algo (n)
if (n>0)
{
    - --
    - --
    - --
    return Algo($\frac{n}{2}$)
}
```

Recurrence Relation :

A recursive function performs a tasks in part by calling itself to perform the subtasks.
At some point, the function encounters a subtask that it can perform without calling itself.
This case, where the function does not recur, is called the base case.

```
A(n)  ________________  T(n)
{
  if (n> 1)  __________  1
    return A(n-1);  ____ T(n-1)
}
```

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

```
function (int n) ———— T(n)
{
   if (n==1) ———— 1
   return; ———— 1
   else
   {
      for (int i=1 ; i <=n ; i++)         }
         for (int j=1 ; j<=n ; j++)       } — n^2
            print ("*")                   }
      function (n-3); ———— T(n-3)
   }
```

$$T(n) = \begin{cases} 1 & , n=1 \\ n^2 + T(n-3) & , \text{otherwise} \end{cases}$$

Methods to solve Recurrence Relation

1. Substitution Method / Backward Substitution Method
2. Recursion Tree Method
3. Master Method.

Follow Now

Start Practicing

i._am._arfin

Arfin Parween