L28
Recursion: Time & Space Complexity

$$n! = \underrightarrow{(n-1)! * n}$$

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

RECAP

$$x^n = x^{n-1} * x$$

$\longrightarrow$ $x^{n/2} * x^{n/2}$ n is even

$x^{n/2} * x^{n/2} * x$ n is odd

Let’s warm up

num % 3 == 0   divisible by 3.

$\underline{num} = num/3$

↳

Check if a given number is a power of 3 or not

27 % 3 = 0

27 / 3 = 9

9 % 3 = 0

9 / 3 = 3

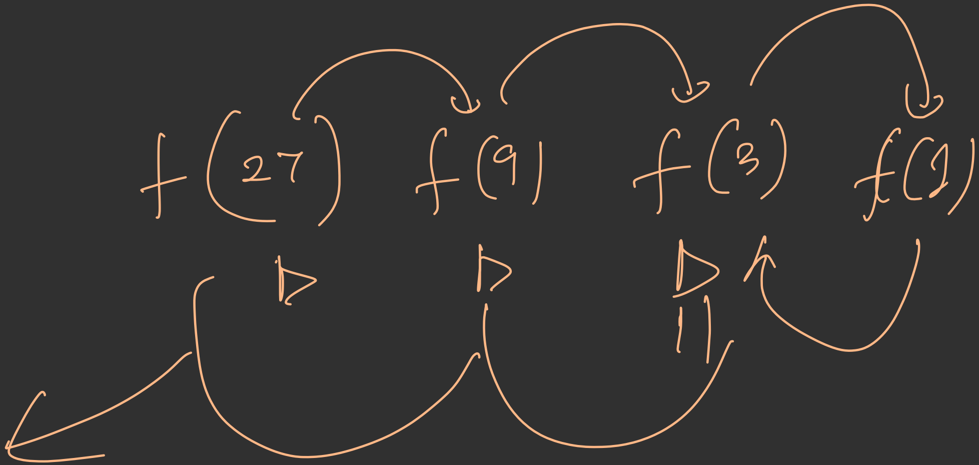
3 % 3 = 0

3 / 3 = 1

$$\begin{tikzcd}
27 \arrow[r, bend left] & 9 \arrow[r, bend left] & 3 \arrow[r, bend left] & 1 \\
3^3 & 3^2 & 3^1 & 3^0
\end{tikzcd}$$

81 % 3 = 0

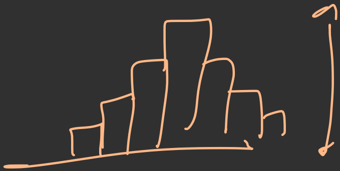
$$\begin{tikzcd}
81 \arrow[r, bend left] & 27 \arrow[r, bend left] & 9 \arrow[r, bend left] & 3 \arrow[r, bend left] & 1 \\
3^4 & 3^3 & 3^2 & 3^1 & 3^0
\end{tikzcd}$$


f(27)
f(9)
f(3)
f(1)

Recursion Tree for the problem we just solved

$$\begin{tikzcd} f(n) \arrow[d] \\ f(n/3) \arrow[d] \\ f(n/9) \\ f(n/27) \end{tikzcd}$$

$$\begin{tikzcd} f(27) \arrow[d] \\ f(9) \arrow[d] \\ f(3) \arrow[d] \\ f(1) \end{tikzcd}$$



4

space → log n

time log n

$n \quad \frac{n}{3} \quad \frac{n}{9} \quad \frac{n}{27} \quad \cdots \quad 1$

$\frac{n}{3^0} \quad \frac{n}{3^1} \quad \frac{n}{3^2} \quad \frac{n}{3^2} \quad \cdots \quad \frac{n}{3^k}$

$1 = \frac{n}{3^k} \qquad 3^k = n \qquad k \; \frac{\log n}{\log 3}$

$k \log 3 = \log n$

Another set of examples

Code 1

```
int power(int a, int n) {
    if(n == 0)
        return 1;
    int partial = power(a, n/2);

    if(n%2 == 0)
        return partial * partial;

    return partial * partial * a;

}
```

Code 2

```
int power(int a, int n) {
    if(n == 0)
        return 1;

    if(n%2 == 0)
        return power(a, n/2) * power(a, n/2);

    return power(a, n/2) * power(a, n/2) * a;

}
```

1. $a^n \rightarrow \boxed{a^{n-1}} * a$

2. $a^n \rightarrow \left. \begin{array}{l} a^{n/2} * a^{n/2} \\ a^{n/2} * a^{n/2} \times a \end{array} \right\}$

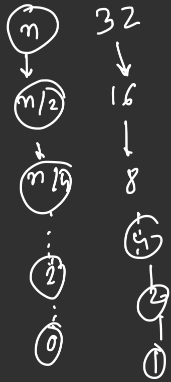
Code 1

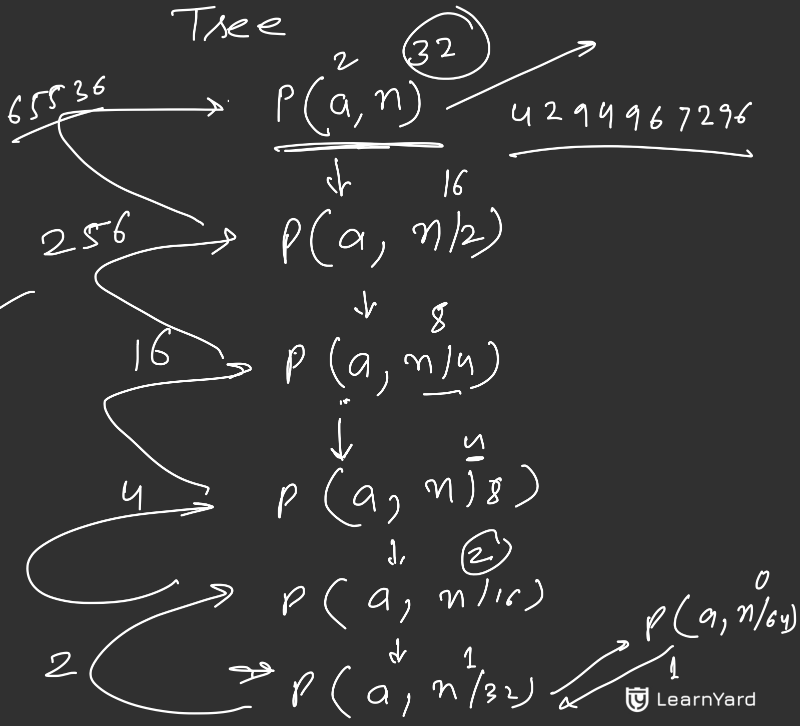
17 ✓

a^{17}

```
int power(int a, int n) {
    if(n == 0)
        return 1;
    int partial = power(a, n/2);

    if(n%2 == 0)
        return partial * partial;
    return partial * partial * a;
}
```

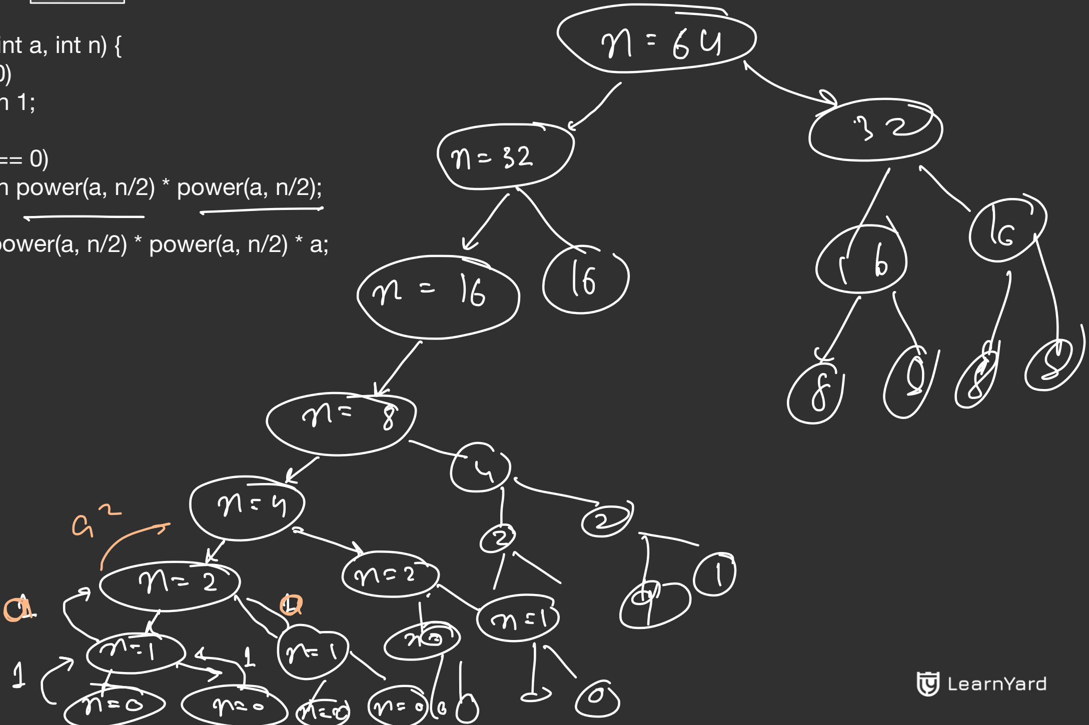
✓

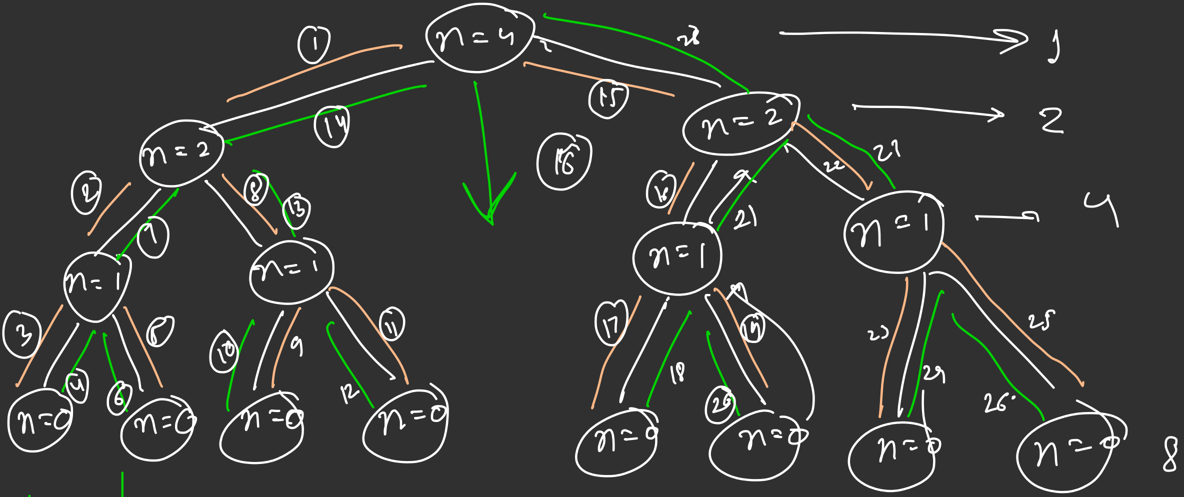


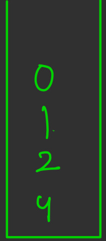


Code 2

```
int power(int a, int n) {
    if(n == 0)
        return 1;

    if(n%2 == 0)
        return power(a, n/2) * power(a, n/2);

    return power(a, n/2) * power(a, n/2) * a;

}
```





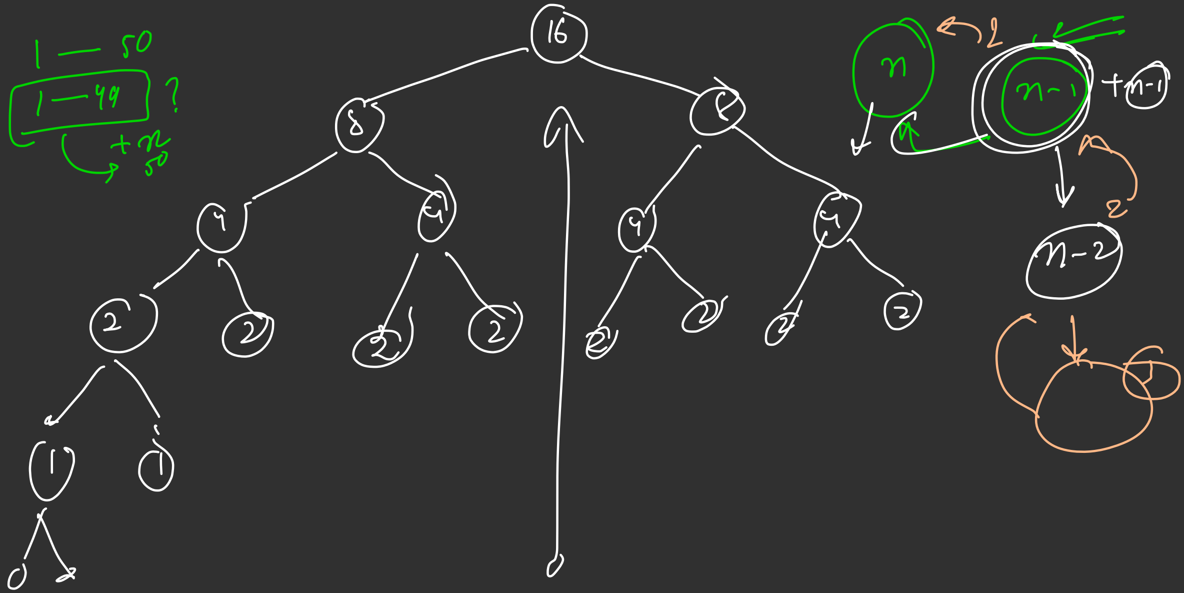


1 2 4 8 16 32 64 . . . . –

log n term

$2^0 \quad 2^1 \quad 2^2 \quad 2^3 \quad 2^4 \ . \ . \ . \ \_ \ \_ \ \_ \ \_$

$$a\left(\frac{r^{\boxed{n}}-1}{r-1}\right) \quad 1\left(\frac{2^{\log_2 n}-1}{2-1}\right) = O(n)$$


1 — 50
1 — 49 ?
+ n
50
16
8
4
2
1
0
n
2
n-1
+ n-1
n-2
2

Recursion Tree 1

Recursion Tree 2

Quick Tip: Time Complexity

Time Complexity in case of recursive code can be calculated by:

(No. of recursive calls)
x
(No. of operations in each recursive call)

Quick Tip: Space Complexity

Space Complexity in case of recursive code:

(Depth of recursion)
x
(Space used in each recursive call)

A different way to analyse : Recurrence Relations

Example 1 : power(a, n)

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

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

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

$$\vdots$$

Example 2 : factorial(n)

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

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

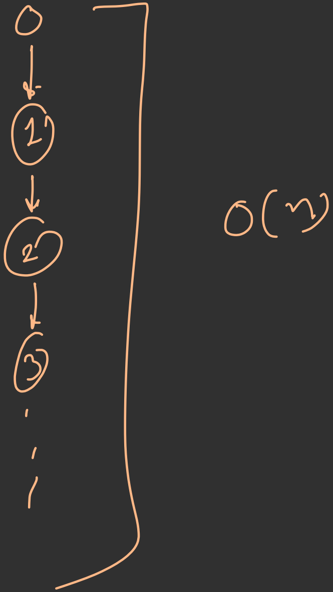
$$+1 \quad \cancel{T(n-2)} = T(n-3) + \boxed{1}$$

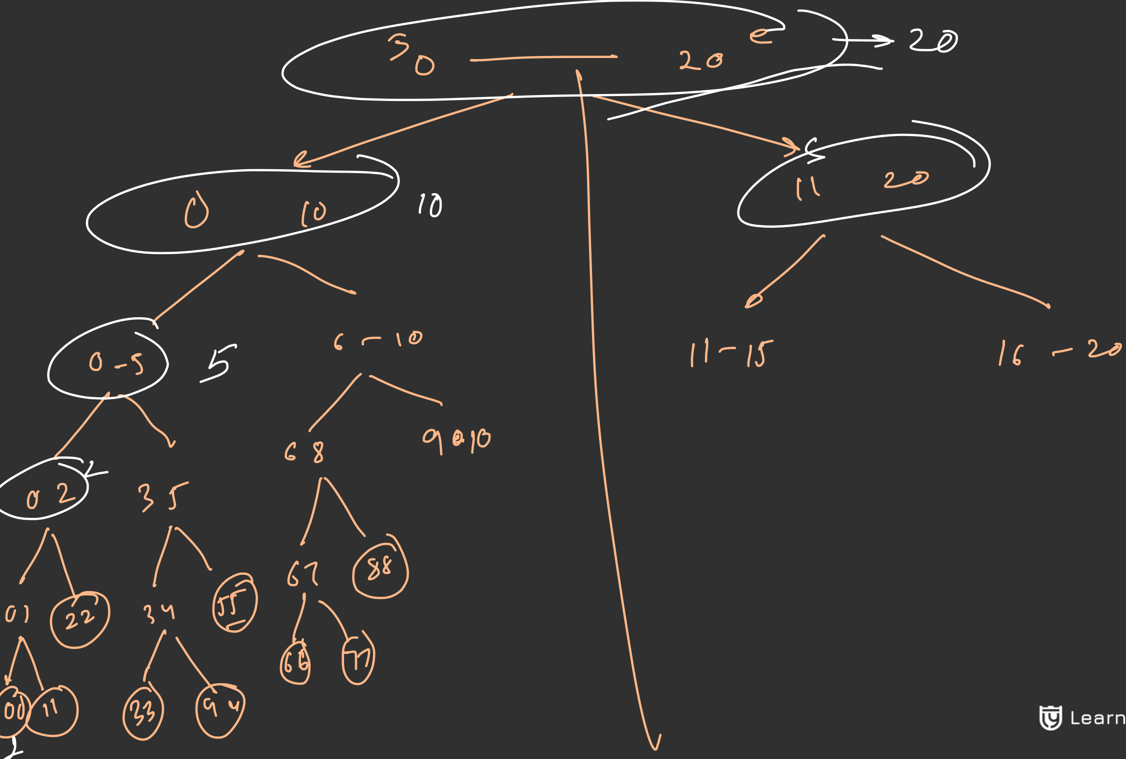
$$\boxed{n} \quad 1 \quad T\,n = \underline{n}$$

Enough Time & Space Complexity now.
Let’s solve a problem?

Given an array and a target K, find the first occurrence of the target inside the Array. Return -1 if K isn’t present inside the Array.

The catch: We’re not allowed any kind of loops (i.e. for, while, do-while are not allowed)


0
1
2
3
O(n)


0 — 20
20
0 10
10
11 20
0-5
5
6-10
11-15
16-20
0 2
3 5
6 8
9 10
0 1
22
3 4
55
6 7
88
00
11
33
44
66
77

Thank You!

Reminder: Going to the gym & observing the trainer work out can help you know the right technique, but you'll muscle up only if you lift some weights yourself.

So, PRACTICE, PRACTICE, PRACTICE!