$F_2 = F_1 + F_0$

don't fall be

Recursive Fibonacci: $F_n = F_{n-1} + F_{n-2}$; ~~n ≥ 0~~ $n \geq 2$; $F_1 = 1$, $F_0 = 0$

Function

~~Termination~~ Termination Condition

if not: Infinite Loop

$F_5 = F_4 + F_3$

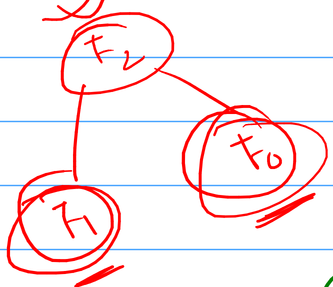
$= F_3 + F_2 + F_2 + F_1$

$= F_2 + F_1 + F_1 + F_0 + F_1 + F_0 + F_1$

$= F_1 + F_0 + F_1 + F_1 + F_0 + F_1 + F_0 + F_1$

$\cancel{= F_0 + F_{-1} + F_{-1} + F_{-2}}$

$= 5F_1 + 3F_0 \notin \mathbb{Z}$




```
fib(n){
    if(n<=0) return -1;
    else{
        if(n<2) return n;
        else return fib(n-1)+fib(n-2);
    }
}
```

If n==0; Return

If n==1; ret 1

Implicit : Abstract : Rough

Explicit : Tacit : Detailed

$f(-1) = -1$

$n=0,\ n=1,\ n=-1$

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

$f(1) = 1$

$f(0) = 0$

$f(-1) = \times \times$

$F_{241} = 20$ years

Let $T(fib(n)) = T(n)$

36 years

```
fib(n){
    if (n<0) ret -1;
    if (n=0) ret 0;
    if (n=1) ret 1;
    return fib(n-1) + fib(n-2);
}
```

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

Cost	Time
c_1	1
c_2	1
c_3	1
c_4	$T(n-1) + T(n-2)$

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

$\therefore T_n = O(2^n)$

Exponential

$O(1)$ = Constant

~~$O(\log\log n)$~~ = Logarithmic

$O(\log n)$ = "

$O(n)$ = Linear

$O(n\log n)$ = Linearithmatic

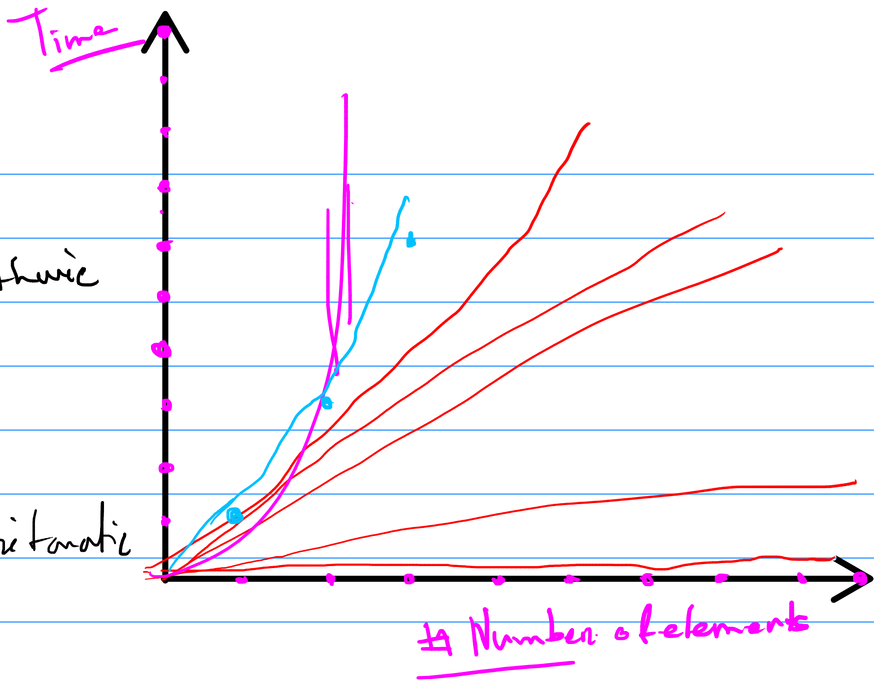
$O(n^2)$ = Qua

$O(n^3)$ = Cu

$O(n^k)$ = Polynom

$O(n!)$

$O(k^n)$ = Exponention $2^n = e^x$



$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
16
32
64
128
256
512
1024
2048

$T_{20} = 2^{20}$ | $\frac{2^{26}}{10^8}$ =

$O(2^{20})$

2^{100}

```
fib(n){
a=0;b=1;tn=0;
for(i=2;i<=n;i++)
      tn=a+b;
      a=b;
      b=tn;
}
```

→ 1
→ $(n-2)$
→ $(n-2) \times 2$
→ $(n-2)$ 1
$(n-2)$ 1

$T(n) = 3 + (n-2)3$
$= 3 + 3n - 6$
$= 3n - 3$
$= 3(n-1)$

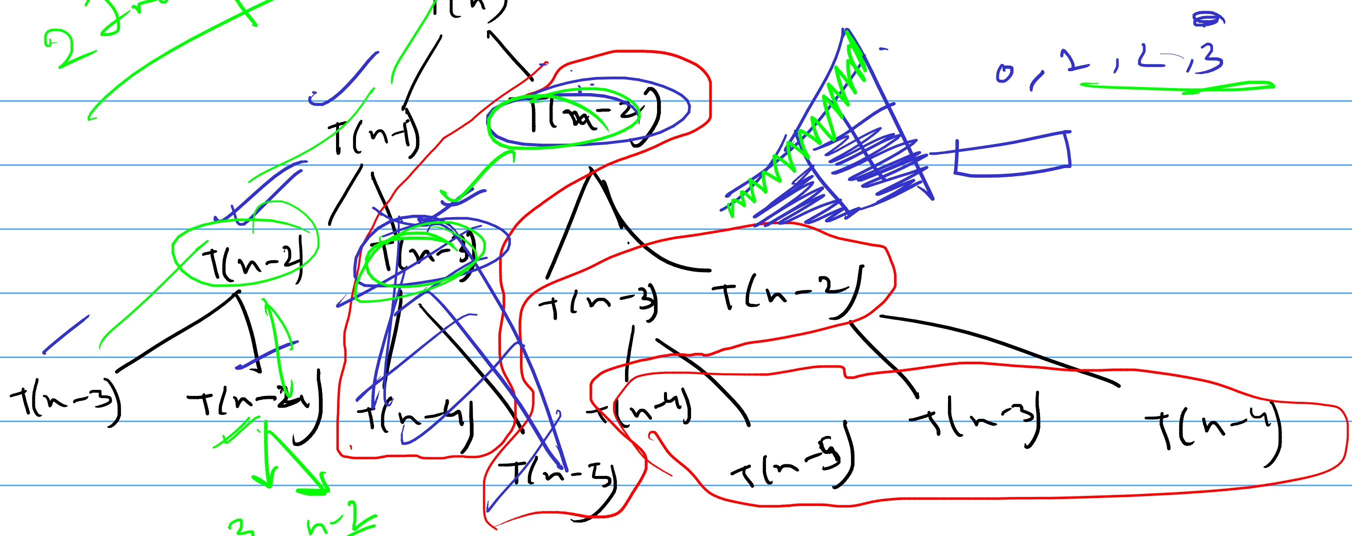
$T(100) = 3 \times 99 = 297$

10^8

2 Index

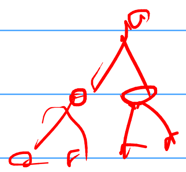
$T_n = T_{n-1} + T_{n-2}$

0, 1, 1, 3



Tabulation =

memorization =



$2^k \rightarrow 1$

n	T(n)
0	0
1	1
2	1
3	2
4	3

Tabulation

Time

~~Memo~~ Memoization

mentaining

Memoes

Memo

Space

n = { , }

T(n) = {

Space

$T(n) = \boxed{\;|\;}$

time

Catalan Numbers:

Tabulation memorization

1232533
2243355
1212412

Suppose we have an 1D array of n numbers. Find a peak T[i] in that array, such that T[i] is greater than it's adjacent elements. Write a linear solution and a DNQ solution. O(N). O(logN). Write another solution to find the peak from a 2D array. O(n^2). O(NlogN)

8,2,5,3,16,2,6,26,22,11,66,24,66