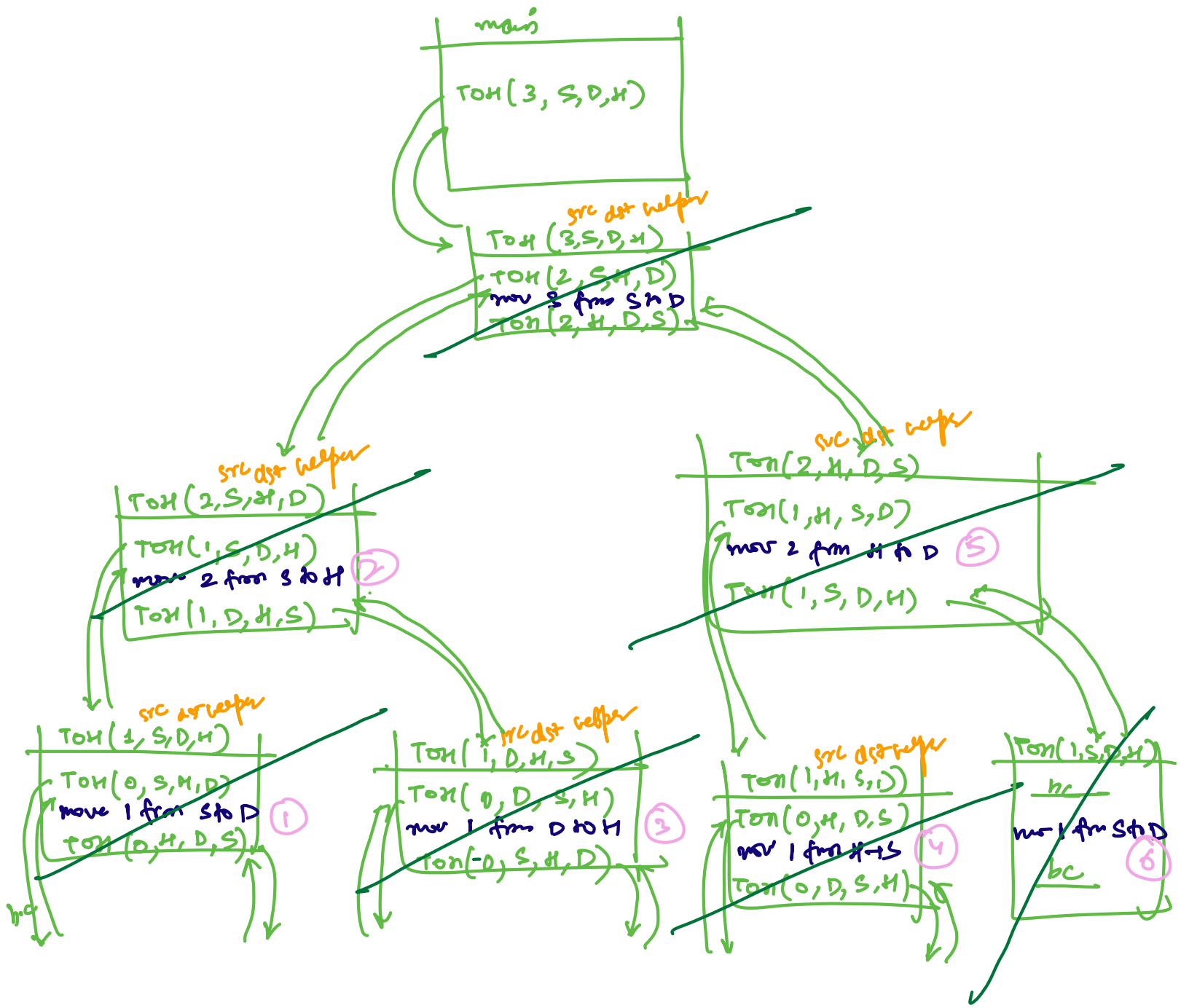
```
void TOH(int n, string src, string dst, string helper)
{
    if(n == 0)
        return ;

    TOH(n-1, src, helper, dst) ;
    cout << "move " << n << " disc from " << src << " to " << dst << endl ;
    TOH(n-1, helper, dst, src) ;
}
```



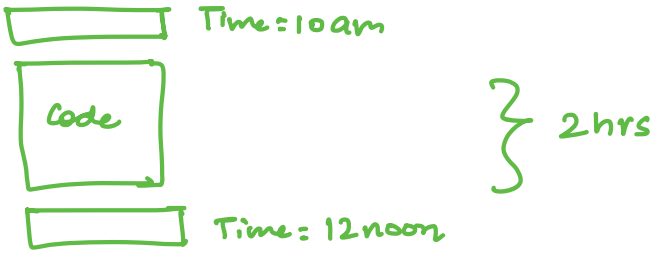
Time complexity

Problem

Least time
Least space

A1 A2 A3 .... An

Experimental Approach:



normal laptop → 2hrs

GPU → 5mins

Environmental conditions / Computation Power

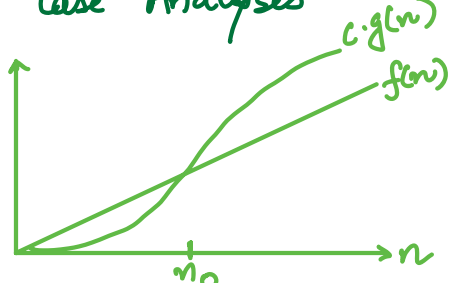
Asymptotic Analysis

↳ how your algo is dependent on the size of input.
n

Asymptotic

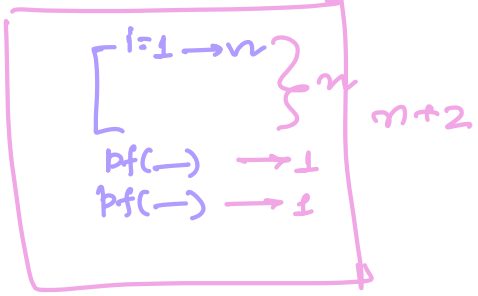
- Worst Case Analysis (Big oh) O
- Best Case Analysis (omega) Ω
- Average Case (Theta) Θ

Worst Case Analysis



$f(n) \leq c \cdot g(n) \quad \forall\ n \geq n_0 \ \&\ c > 0$

$f(n) = O(g(n))$



Eg. $f(n) = n+2$

Big-Oh of $f(n)$?

$f(n) \leq c \cdot g(n)$

$n+2 \leq c \cdot n$

$n+2 \leq 3 \cdot n$

$n=1$	3	$\leq$	3
$n=2$	4	$\leq$	6
$n=3$	5	$\leq$	9
	⋮		⋮

$n+2 \leq 3 \cdot n \quad \forall\ n \geq 1$ & $c=3$

$f(n) \quad c \quad g(n)$

$n+2 = O(n)$

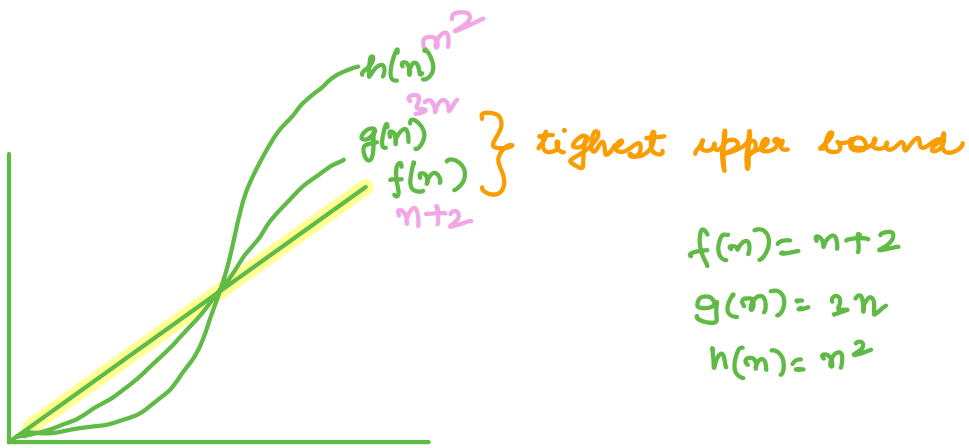
Eg: $f(n) = 2n^2 + 3n + 1$

Big Oh of $f(n)$ = ?

$2n^2 + 3n + 1 \leq 6 \cdot n^2$

$n=1$	6	$\leq$	6
$n=2$	8+6+1	$\leq$	24
	15	$\leq$	24

$2n^2 + 3n + 1 = O(n^2)$



$f(n) = n+2$

$g(n) = 3n$

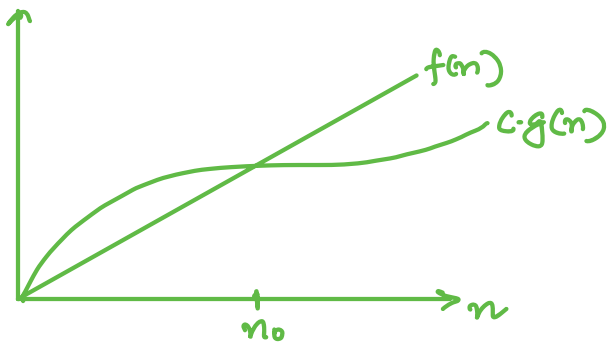
$h(n) = n^2$

```
i: 1 ——— n
    j: 1 ——— n
Pf(hello)
Pf(bye)
i: 1 ——— n
```

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

$= O(n^2)$ ✓ W.C.

Best Case Analysis



$f(n) \geq c \cdot g(n) \quad \forall\ n \geq n_0 \ \&\ c > 0$

$f(n) = \Omega(g(n))$

Linear Search

0	1	2	3	4
10	20	5	100	50

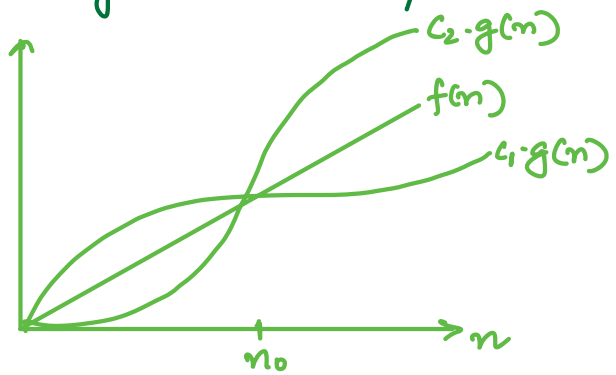
$n = 5$

$BC = \Omega(1)$

$WC = \underbrace{n-1}_{f(n)} \leq \underbrace{1 \cdot n}_{c \ \ g(n)} \qquad O(n)$

Average Case Analysis

WC & BC both are same



$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$

$f(n) = \Theta(g(n))$

Time Complexity Q's

1.
```
i=1
while (i≤n)
{
    pf(DTU);
    i++;
}
```
O(n)

2.
```
i=0
while (i≤n)
{
    pf(DTU)
    i+=2;
}
```
$\frac{n}{2}$ O(n)

3.
```
i=1
while (i≤n)
{
    pf(DTU)
    i=i+2;
    i=i+3;
}
```
i = i+5

$\frac{n}{5}$ O(n)

4.
```
i=1
while (i≤n)
{
    pf(DTU);
    i=i*2;
}
```
k times

time	i value
1	2^1
2	2^2
3	2^3
4	2^4
⋮	
k	2^k

$i > n$

$2^k > n$

$k = \log_2 n$

5.
```
i=1
while (i≤n)
{
    pf(DTU);
    i=i*3;
}
```
$\log_3 n$

6.
```
i=1
while (i≤n)
{
    pf(DTU);
    i=i*2;
    i=i*3;
}
```
i = i*6

$\log_6 n$

7.
```
while (n>0)
{
    pf(DTU)
    n=n/2;
}
```
$\log_2 n$

8.
```
while (n>0)
{
    pf(DTU)
    n=n/3;
}
```
$\log_3 n$

9.
```
while (n>0)
{
    pf(DTU)
    n=n/2;
    n=n/3;
}
```
$\log_6 n$

Nested loops

- inner loop
- outer loop
- independent

```
for (i=1 — n)        → n
{                       *      } n^2
   for (j=1 — n)     → n
   {

   }
}
```

dependent (unrole) → inner most line : no of times execute

```
for (i=1 — n)
{
   for (j=1 — i)
   {

   }
}
```

Q:

```
for (i=1; i<=n; i++)
{
    for (j=1; j<=n; j=j+i)
    {
         pf(DTU);
    }
}
```

i = 1	i = 2	i = 3	. . . .	i = n
j=1 ; j<=n; j=j+1	j=1; j<=n; j=j+2	j=1; j<=n; j=j+3		j=1; j<=n; j=j+n
n times	$\frac{n}{2}$ times	$\frac{n}{3}$		$\frac{n}{n}$ = 1 time

$$n + \frac{n}{2} + \frac{n}{3} + \cdots \frac{n}{n}$$

$$n\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \text{———} \frac{1}{n}\right)$$

$n \log n$

$$\int_{i=1}^{n} \frac{1}{i} = \log i \Big|_{i=1}^{n}$$

$= \log n$

Q: `i=1, S=1`

```
while(i<=n)
{
    pf(DTU);
    S=S+i;
    i++;
}
```

`i=1, i<=n, i++` } O(n)

`i=1, S=0`

```
while(S<=n)
{
    pf(DTU);
    S=S+i;
    i++;
}
```

} k times

i=1 1st time → S=1

i=2 2nd time → S=1+2

i=3 3rd → S=1+2+3

i=k kth → S=1+2+3+....k

S>n

1+2+3 ⟶ k > n

$\frac{k(k+1)}{2} > n$

$k^2 > n$

$k = \sqrt{n}$

Q:

```
for(i=1; i<=k; i++)        ⟶ k
{
    for(j=1; j<=n/k; j++)  ⟶ n/k
    {
        pf(DTU);
    }
}
```

$k * \frac{n}{k} = O(n)$

i++ → 10, 20, 30, 40, 50, 60 ← j--

O(n)

Palindrome Reverse

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

} $\frac{n}{2} = O(n)$

Prime

SOE

n given — Prime ? ✓ ×

↓

$\sqrt{n}$

36

$\sqrt{36} = 6$

2 × 18
3 × 12
4 × 9
6 × 6 → $\sqrt{26}$
9 × 4
12 × 3
18 × 2

2 | $\sqrt{n}$ — $\sqrt{36}$

2 | 6

$n = 10^5$

$1 \rightarrow 10^5$

↓

$n\sqrt{n}$

SOE

not cut : true
cut : false

~~1~~	**2**	**3**	~~4~~	5
~~6~~	**7**	~~8~~	~~9~~	~~10~~
11	~~12~~	**13**	~~14~~	~~15~~
~~16~~	**17**	~~18~~	**19**	~~20~~
~~21~~	~~22~~	**22**	~~24~~	~~25~~

1 → 25 $\sqrt{25} = 5$

2 ✓ … 5

2
3
4 ×
5

n = 25

0 → 25

0 1

2 × 3
3 × 2 ×
3 × 3 ✓

table multiples

2 × 2 = 4
× 3 = 6
× 4 = 8
× 5 = 10
×

Cn