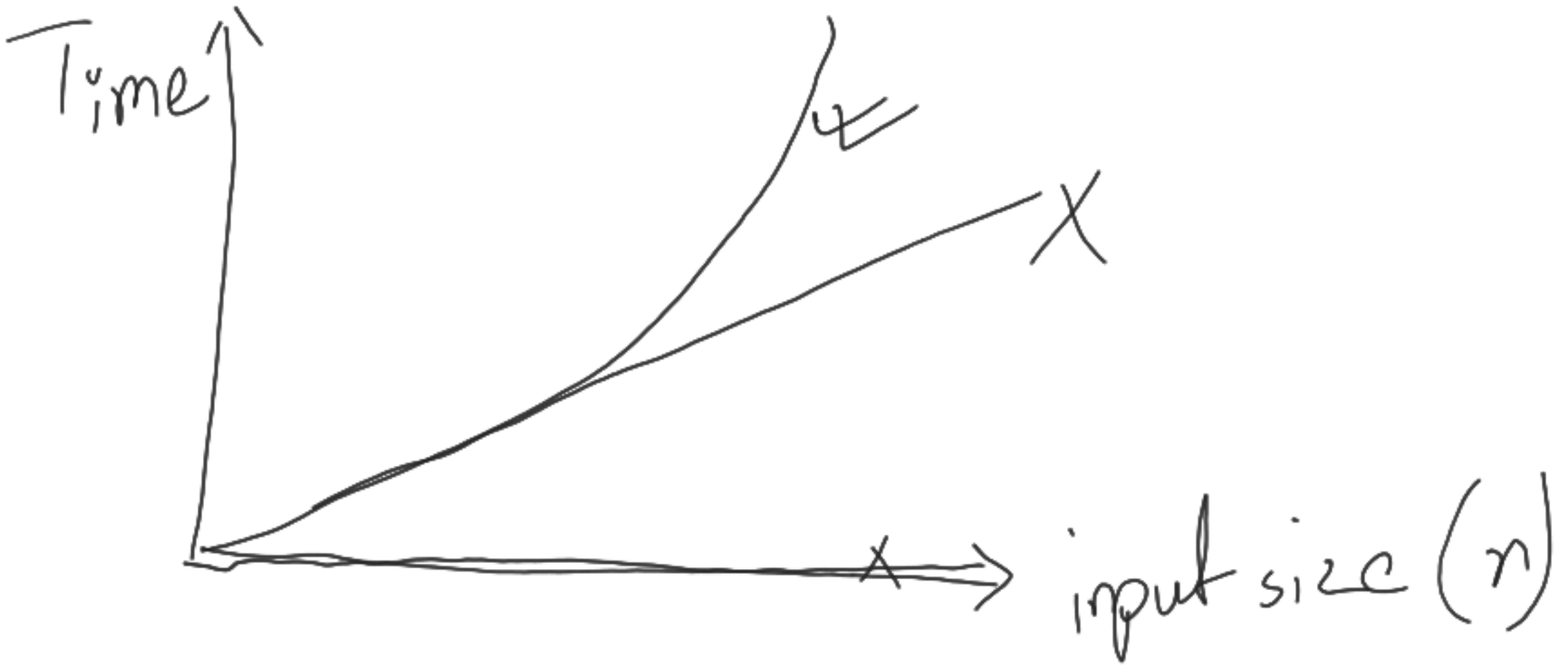
$f(x) = \frac{x^2}{2} + x + 1 \longrightarrow O(n^2)$

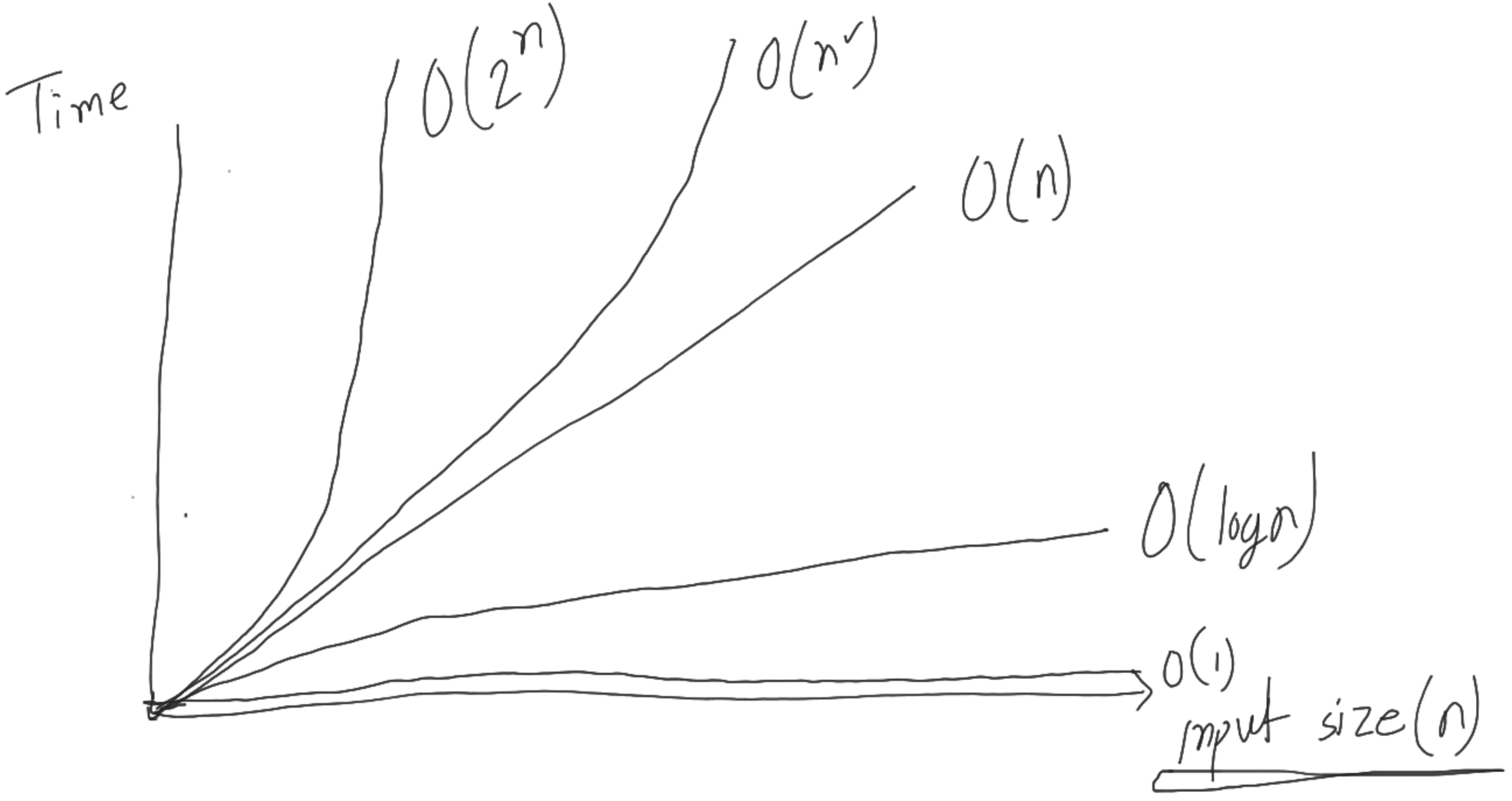


Big O notation spec.

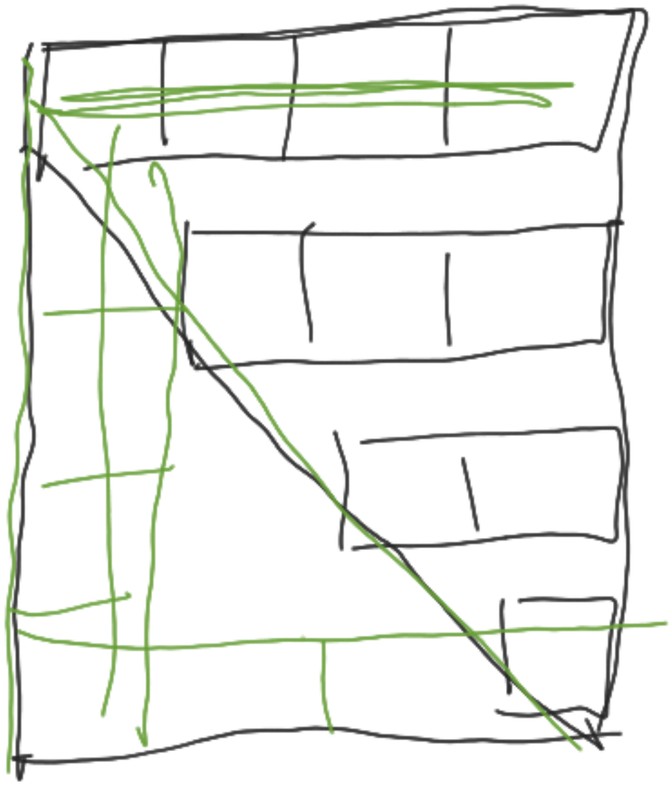
1. Don't consider constants
2. Only consider the max power variable
3. Always consider worst case scenario

$$f(x) = \left(\frac{x+3}{2}\right)^3 + x^2 + 3 \longrightarrow O(x^3)$$

$$f(n) = n\log n + \log n + 10 \rightarrow O(n\log n)$$


Time
O(2^n)
O(n^r)
O(n)
O(log n)
O(1)
input size(n)

[[1, 2],
[1, 3],
[1, 4],
[2, 3],
[2, 4],
[3, 4]]



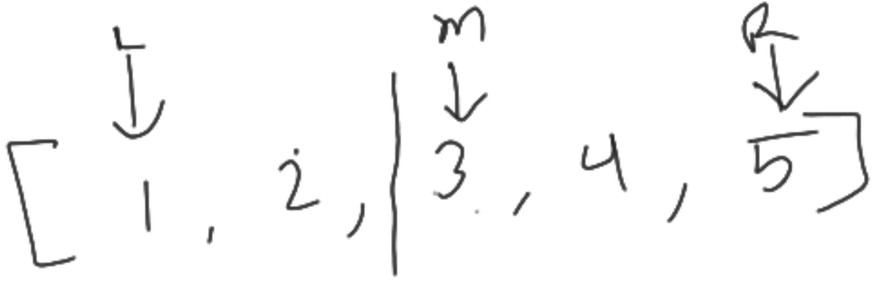
n = 2

c = n

n · n = n^2

$O(n^2)$

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



t = 4

m = (L + R // 2)

while(L <= R)

= 0 + 4 // 2

= 2

t == m → return m

t > m → L = m + 1

t < m → R = m - 1

n = [1 , 2 , 3 , 4 , 5]

L

R

L = 0

R = n.length − 1

m = L + R // 2

if (t == m)

return m

if (t < m)

n = 50 / 2

↓

25 / 2

↓

12 / 2

↓

6 / 2

↓

3 / 2 → 1

log n

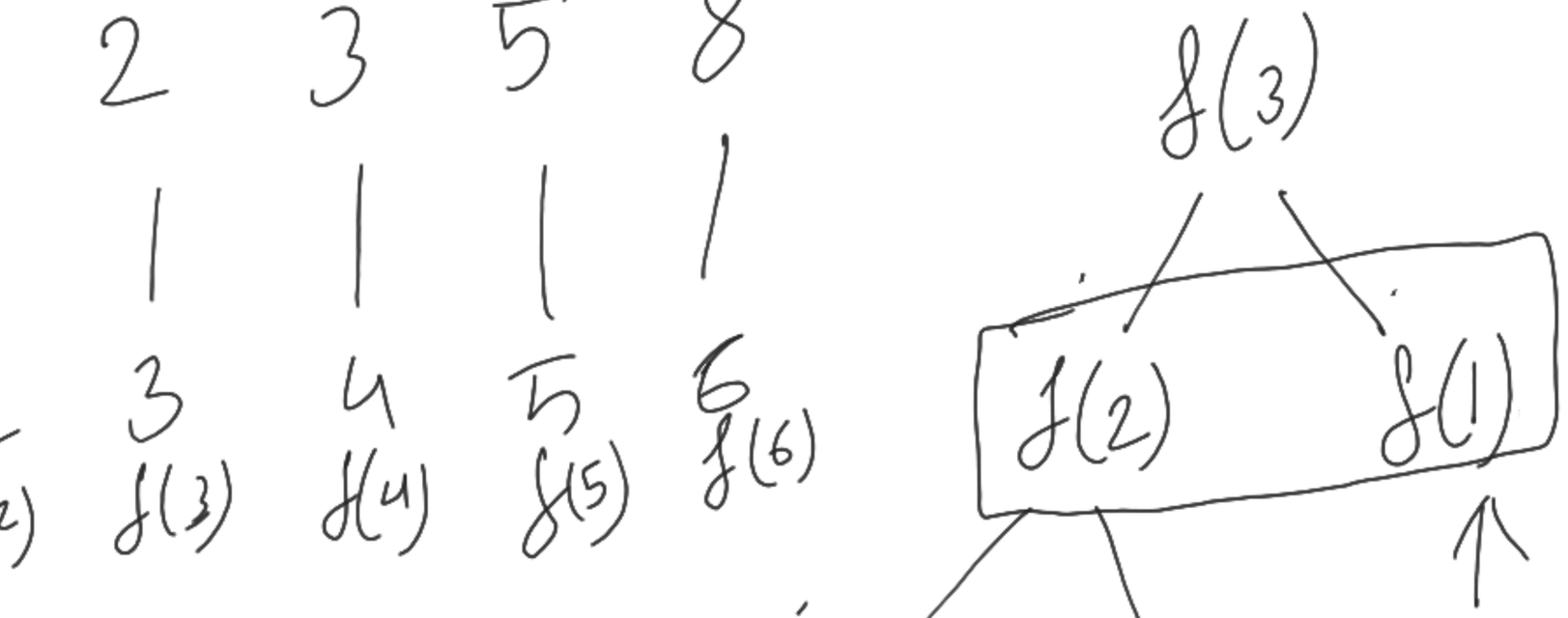
log 50 = 5

0 1 1 2 3 5 8

1 1 1 1 1 1 1

0 1 2 3 4 5 6

f(0) f(1) f(2) f(3) f(4) f(5) f(6)



$n = 10$

$2 \times 2 \times 2 \times 2 \cdots$

$2^{10} = 2^n$