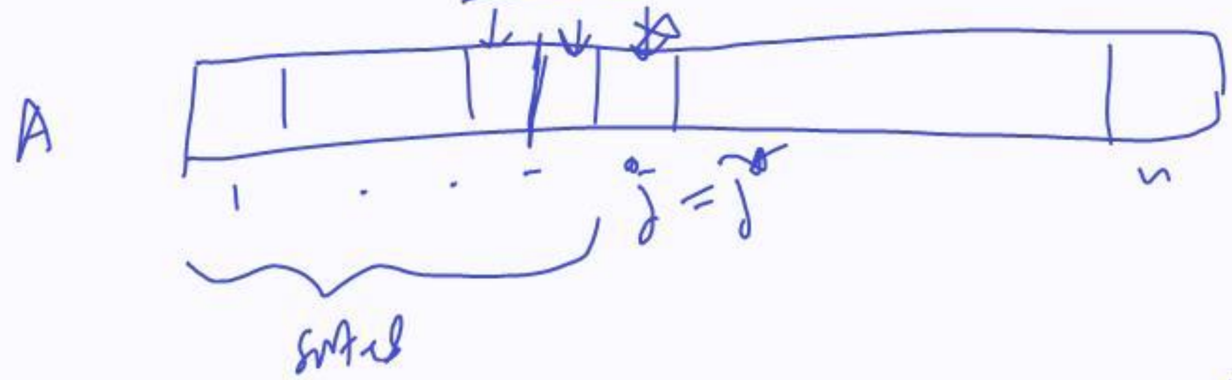
Lecture 2



Claim: After the iteration of outer for loop finishes with j

$A[1] \ldots A[j]$ is sorted

Proof: I.H.. $\underbrace{j = j^* - 1}$ $\Rightarrow j = j^*$

$\{A[1] \ldots A[j^*-1], A[j^*]\}$ sorted after inner loop finishes.

$i = j^*-1, \ldots, 1$

$\square$

Asymptotic time complexity

How much time an algorithm takes?

C

c'

```
Procedure InsertionSort (Array A, size n)
    for j = 2 to n
        for i = j-1 to 1
            if (A[i] > A[i+1])
                // swap A[i] & A[i+1]
            endif
        endfor
    endfor.
```

$\boxed{\text{Assumption}}$: basic operations take constant time

- if
- $A \leftarrow B$
- $\vdots$

$(n-1)$: $j = 2$ to n

$i = j-1$ to 1

$(c + 3c')$

$\Downarrow$

$$\sum_{j=2}^{n} (j-1) = \frac{(n-1)n}{2} \quad [\;\Downarrow\;]$$

$$\underline{\Theta(n^2)} \Longleftarrow \approx \underline{d} \cdot n^2$$

Example:

A_1 : $100n$ time

A_2 : $n^2 + 5n$ time.

$n \to \infty$.

$$n^2 + 5n \leq 100n$$

$$\equiv \quad n \leq 95$$

- ignore small behavior. We'll focus on asymptotic behavior, $n \to \infty$

- ignore any leading constants.

$$100n \approx 3n$$

Definition [Big Oh Notation].

A function $f(n) = \boxed{O(g(n))}$ if

$\exists$ two constants c & d such that

$$f(n) \leq \boxed{c}\, g(n)$$

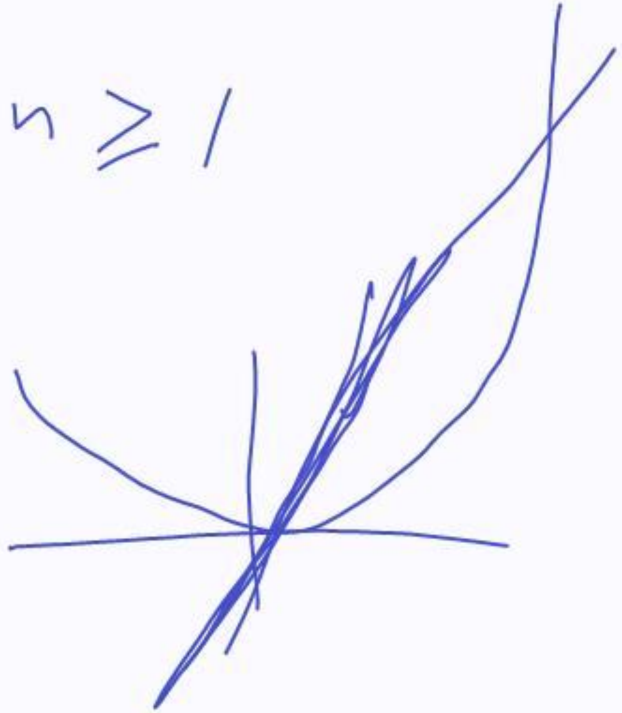
for all $n \geq d$

- ignoring constants.
- ignoring small behaviour.

(1) $100n = O(n^2 + 5n)$?

Yes. $c = 100, d = 1$

$100n \le 100(n^2 + 5n) \quad \forall n \ge 1$



(2) $n^2 + 5n = O(n^2)$.

Yes : $c = 6$

(3) $n^2 = O(n)$?

No! $\nexists\ c, d$ s.t

$n^2 \le c \cdot n \quad \forall\ \underline{n \ge d}$.

Equivalently: $\lim_{n \to \infty} \frac{f(n)}{g(n)}$.

- if this limit exists & $\leq c$ for some constant c

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

① $f = 100n$, $n^2 + 5n = g(n)$.

$$\lim_{n \to \infty} \frac{100n}{n^2 + 5n} \to 0.$$

② $f(n) = n^2$, $g(n) = n$ $\qquad \lim_{n \to \infty} \frac{n^2}{n} \to \infty$

$n^2 \neq O(n)$.

Two more notations:

(1) $f(n) = \underbrace{\Omega}_{\text{Omega}}(g(n))$. iff $g(n) = O(f(n))$.

Example $f(n) = n^2$, $g(n) = 100n$

~~100n =~~ $g(n) = O(f(n))$.

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

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

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

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

$$f(n) = O(g(n)) \equiv \leq$$

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

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

Exercise:

(1) $c \cdot n^2 = O(n^2)$ $\forall$ constant c

(2) $n^b = O(n^a)$ $\forall$ $a \geq b$

(3) $n^a = O(2^n)$ for any constant a

(4) $(\log n)^a = O(n^b)$ for any constants a, b.

$\log n = O(\sqrt{n})$.

big O, Ω, Θ $\boxed{n \geq 1, \text{integer}}$

O o

little o , ω Two more notations

(1) $f(n) = o(g(n)) \equiv <$

little oh

$f(n) = n^2$, $g(n) = 2n^2 + 3n$

$f(n) = o(g(n))$ iff $f(n) \neq \Omega(g(n))$

little oh

(2) $f(n) = \omega(g(n)) \equiv >$

iff $f(n) \neq O(g(n))$

A quick test

(if this exists) $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$.

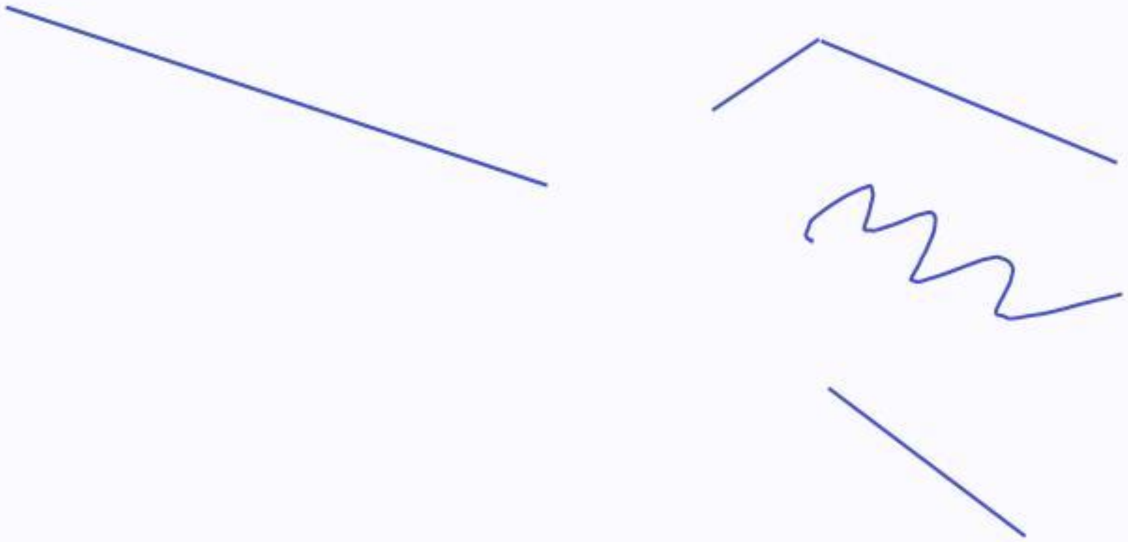
if $c = 0$ $\quad f(n) = o(g(n))$

if $c = \text{constant}$ $\quad f(n) = O(g(n))$

if $0 < c < \infty$ $\quad f(n) = \Theta(g(n))$.

if $c = \infty$ $\quad f(n) = \cancel{\neq}\ \omega(g(n))$.

if $c \neq 0$ $\quad f(n) = \Omega(g(n))$



Fibonacci Sequence

0, 1, 1, 2, 3, 5, 8, 13, . — —

$F_0 = 0$

$F_1 = 1$

$F_n = F_{n-1} + F_{n-2}$.

"Pingala"

Recursive vs Iterative algorithm.

Procedure RFib (n).
// returns the n-th Fibonacci number.
if (n = 0) Return 0 end if
if (n = 1) Return 1 end if
Return (RFib (n-1) + RFib (n-2))

Correctness ?

Time complexity :