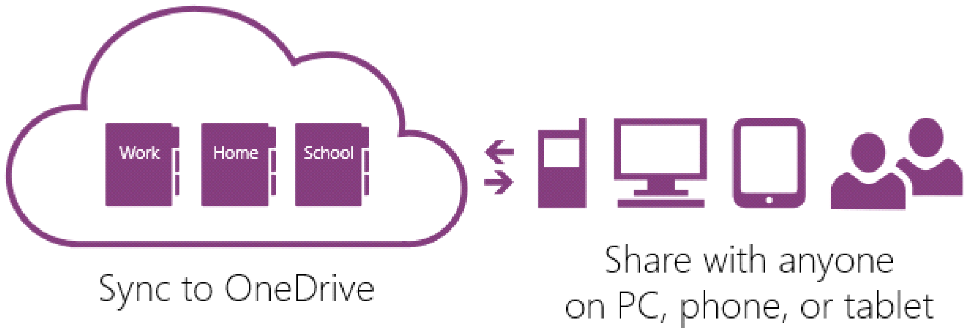
OneNote: one place for all of your notes



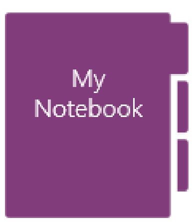
[Watch the 2 minute video]

1. Take notes anywhere on the page

Write your name here

2. Get organized

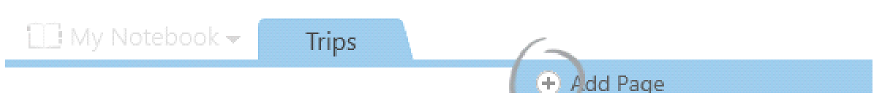
You start with "My Notebook" - everything lives in here

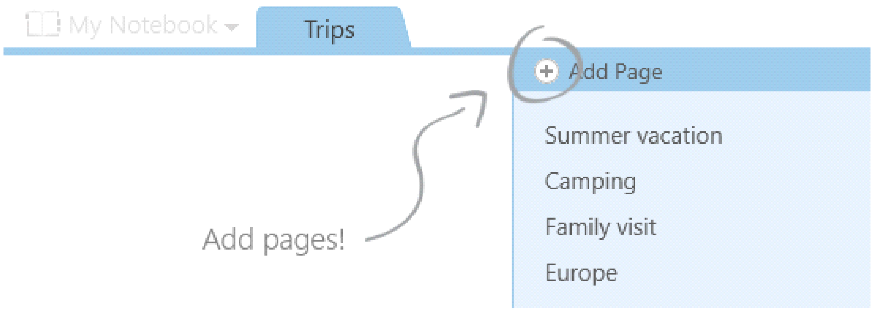


Add sections for activities like:

Add sections!

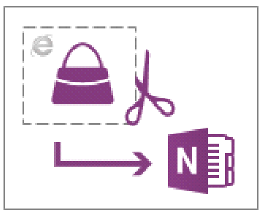
Add pages inside of each section:



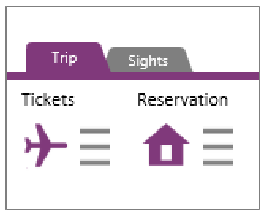


(Pages are over there)

3. For more tips, check out 30 second videos



Clip from the web



Plan a trip with others

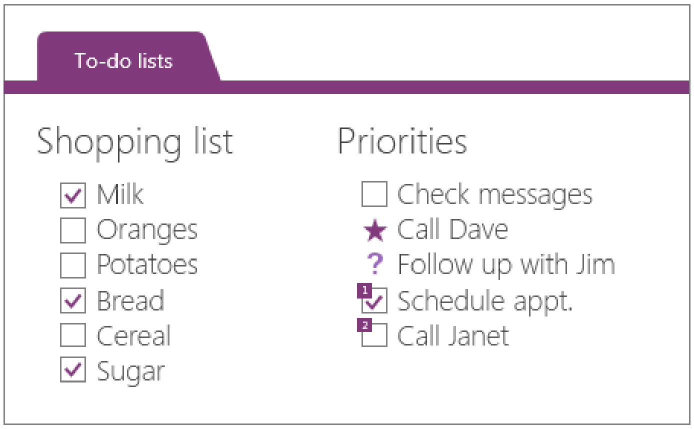
Search notes instantly

Write notes on slides

4. Create your first page

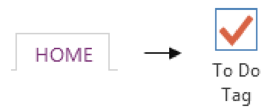
You're in the Quick Notes section - use it for random notes

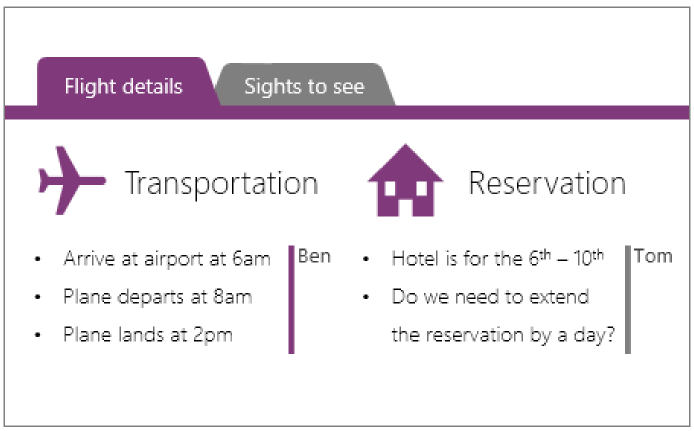
OneNote Basics



Remember everything

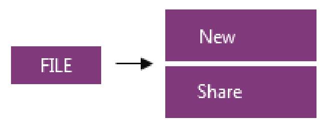
- Add Tags to any notes
- Make checklists and to-do lists
- Create your own custom tags





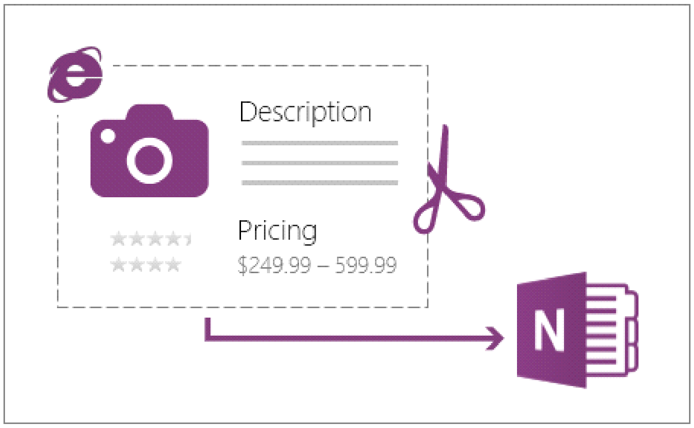
Collaborate with others

- Keep your notebooks on OneDrive
- Share with friends and family
- Anyone can edit in a browser



Keep everything in sync

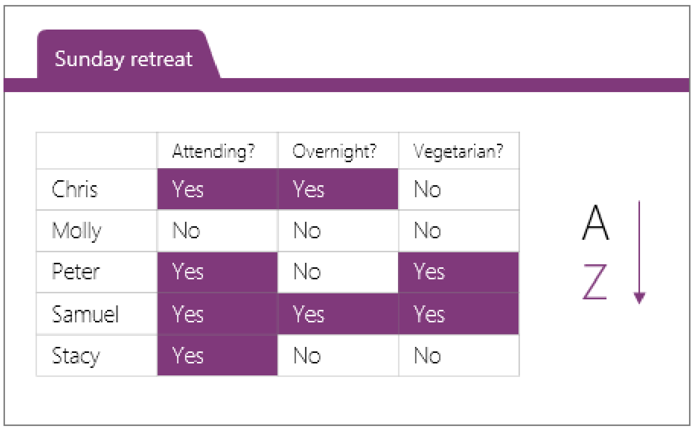
- People can edit pages at the same time
- Real-Time Sync on the same page
- Everything stored in the cloud
- Accessible from any device



Clip from the web

- Quickly clip anything on your screen
- Take screenshots of products online
- Save important news articles

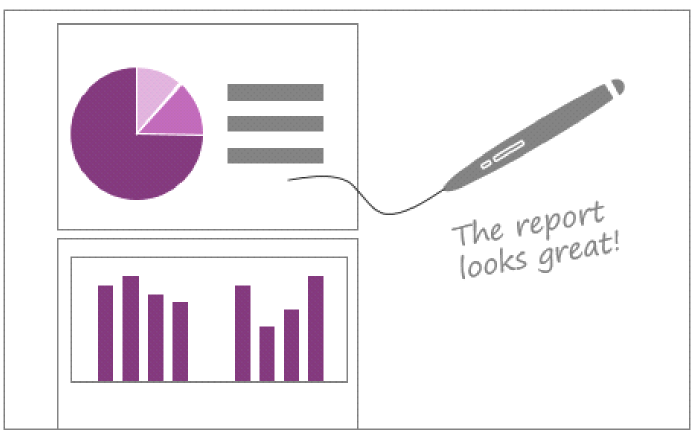
in your taskbar
OR
+ S on your keyboard



Organize with tables

- Type, then press TAB to create a table
- Quickly sort and shade tables
- Convert tables to Excel spreadsheets

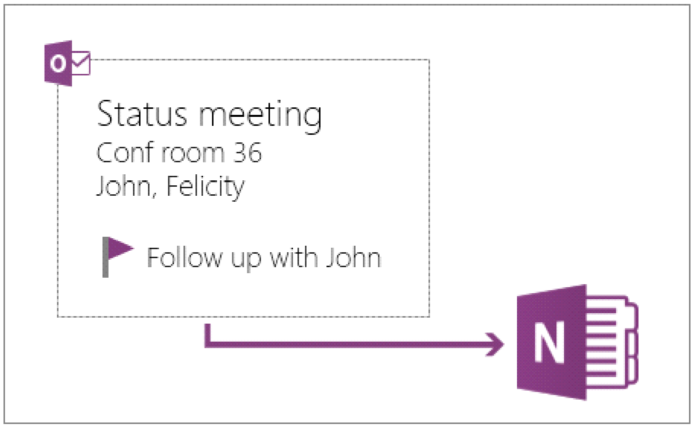
TABLE TOOLS
LAYOUT
Shading
Sort
Convert to Excel Spreadsheet



Write notes on slides

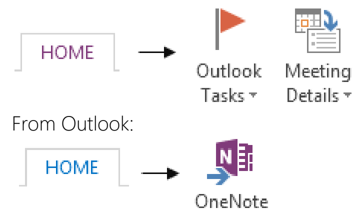
- Send PowerPoint or Word docs to OneNote
- Annotate with a stylus on your tablet
- Highlight and finger-paint

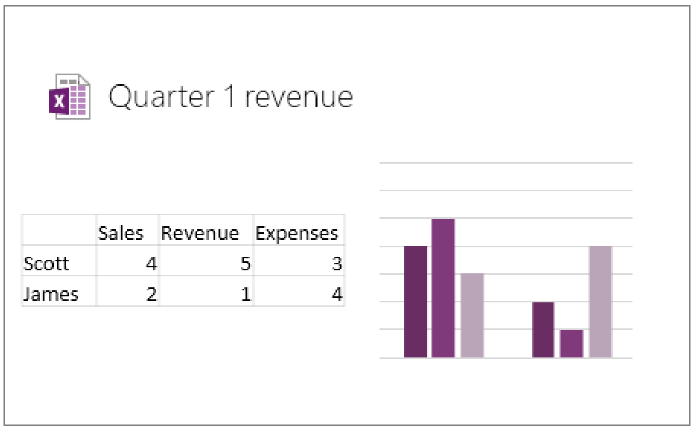
in your taskbar
OR
+ N on your keyboard



Integrate with Outlook

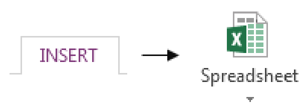
- Take notes on Outlook or Lync meetings
- Insert meeting details
- Add Outlook tasks from OneNote

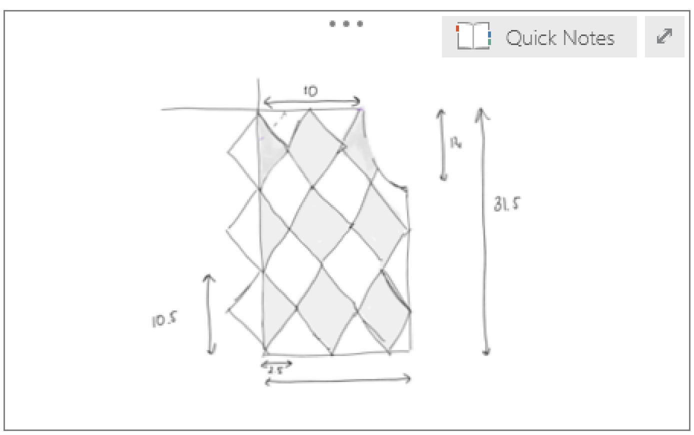




Add Excel spreadsheets

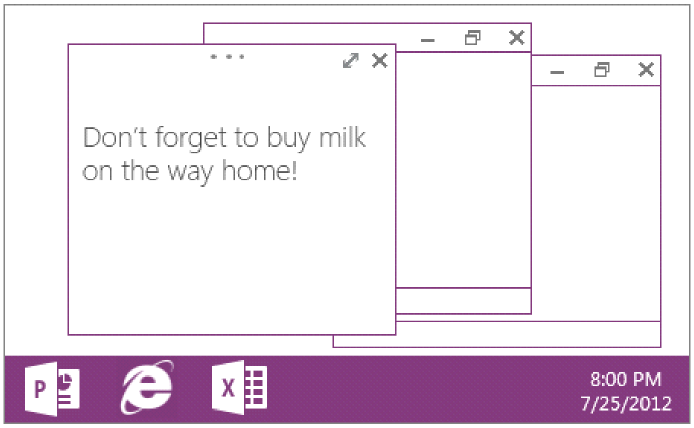
- Track finances, budgets, & more
- Preview updates on the page





Brainstorm without clutter

- Hide everything but the essentials
- Extra space to focus on your notes



Take quick notes

- Quickly jot down thoughts and ideas
- They go into your Quick Notes section

 in your taskbar

OR

 + N on your keyboard

CSE 674 (Data Structures & Algorithms)

Monday, August 29, 2016 6:44 PM

Advanced

→ Andrew Lee.

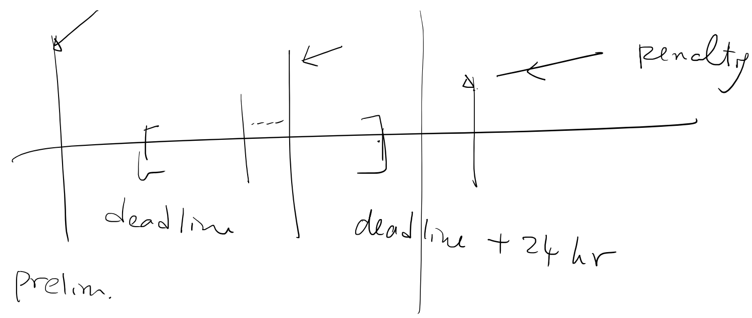
DS + Alg. = Programs

(Wirth)

Organization Meeting

+ preliminaries

24 hr ~~extension~~ applies



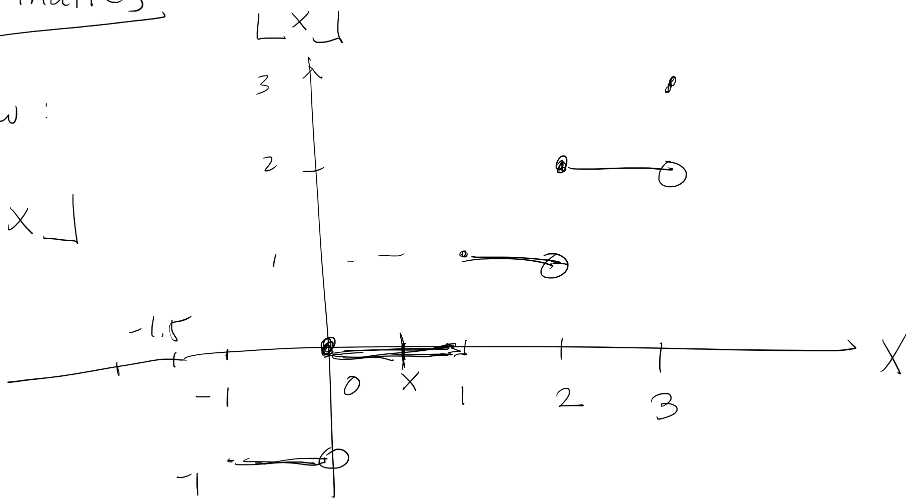
Preliminaries Math Appendix of CLRS

Preliminaries

$\lfloor -1.5 \rfloor = \begin{cases} -1 \\ -2 \end{cases}$

Review:

$\lfloor x \rfloor$



$\longrightarrow$ =

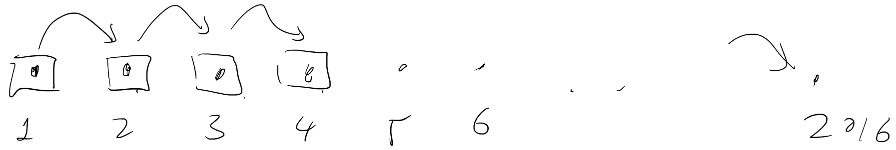
Math Induction $P(n)$: statement to be proved

✓

base case $n = 1$ ✓

Induction Assumption: $P(n)$ Assumes True
show $P(n+1)$ is true. } If $P(n)$ then $P(n+1)$

Then Principle of M.I. ✓
~~shows~~ tells us $P(n)$ is true for all n.



Sets

Multi sets

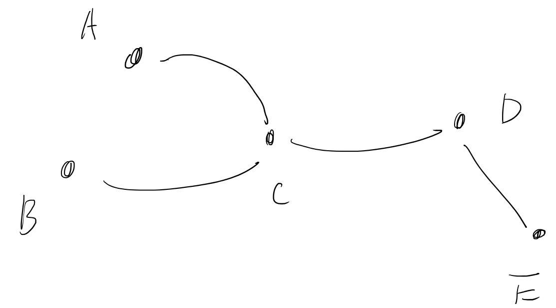
$A = \{1, 2, 3\}$

$B = \{1, 2, 3, 3, 1\}$

$A \stackrel{?}{=} B$ (if A, B are sets) ✓

(if A, B are multi-sets)

Class example



→ Tree

Vertices $= \{A, B, C, D, E\}$

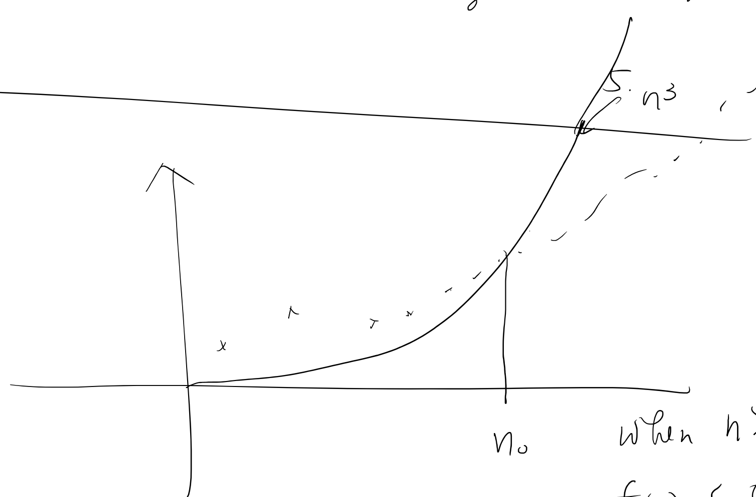
Edges $= \{\{A, C\}, \quad \}$

an edge

remaining edges

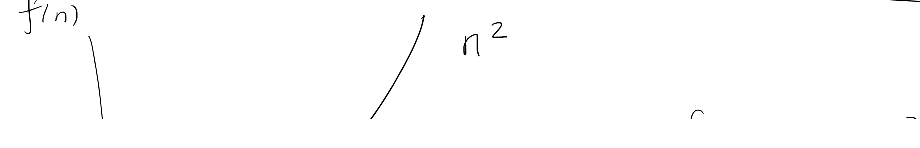
$f(n) = O(n^3)$

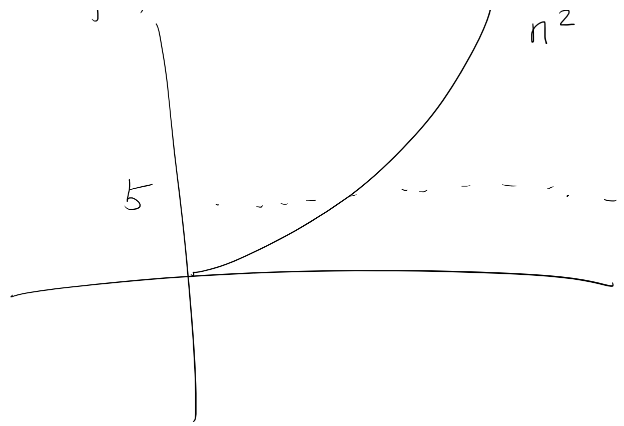
upper bound



when $n \geq n_0$

$f(n) \leq 5 \cdot n^3$





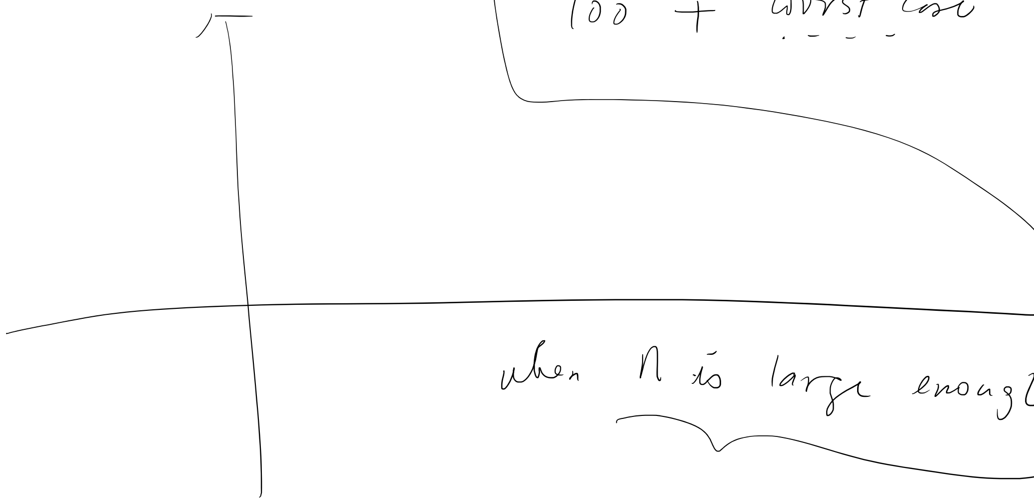
$f(n) = O(n^2)$

$100 + \text{worst case} \le 100 + 5n^2$

$\le 6n^2$

when

$n \geq \ldots$



when n is large enough

Bubble Sort worst case : $O(n^2)$

```
for ( ~ n steps ) {
    for ( ~ n steps) {
    } constant amount of work
}
```

input . 2^n step . $O(n)$

k step $O(\lg_2 k)$

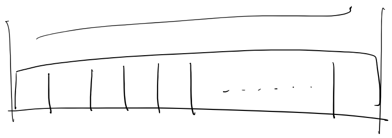
9/7/2016

Wednesday, September 7, 2016 6:43 PM

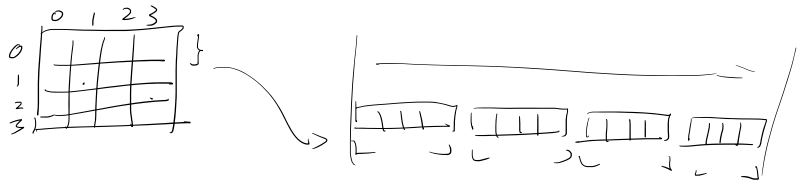
CSE 674

Sequential Allocation

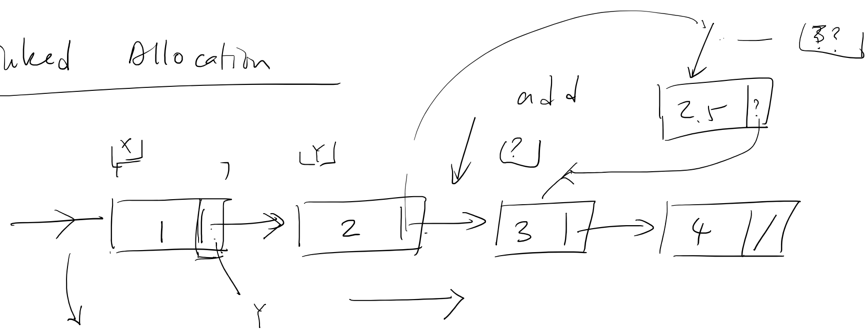
Linked Allocation.

Array.  (1 dim)

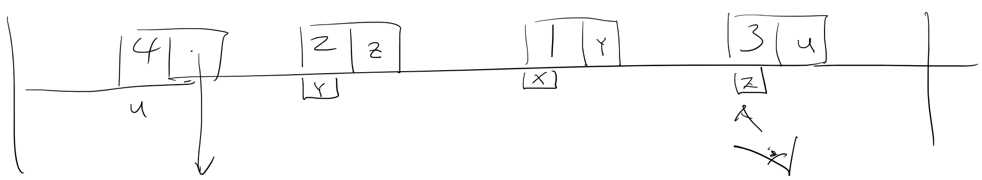
→ high dimension.

2-dim. 

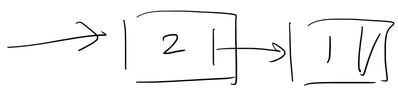
Linked Allocation



address location



pointers.cpp (from phils-ads 02)

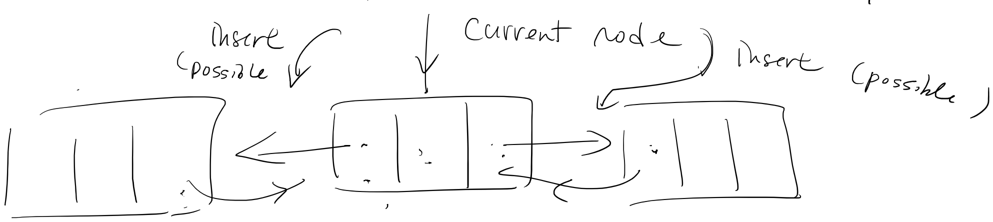


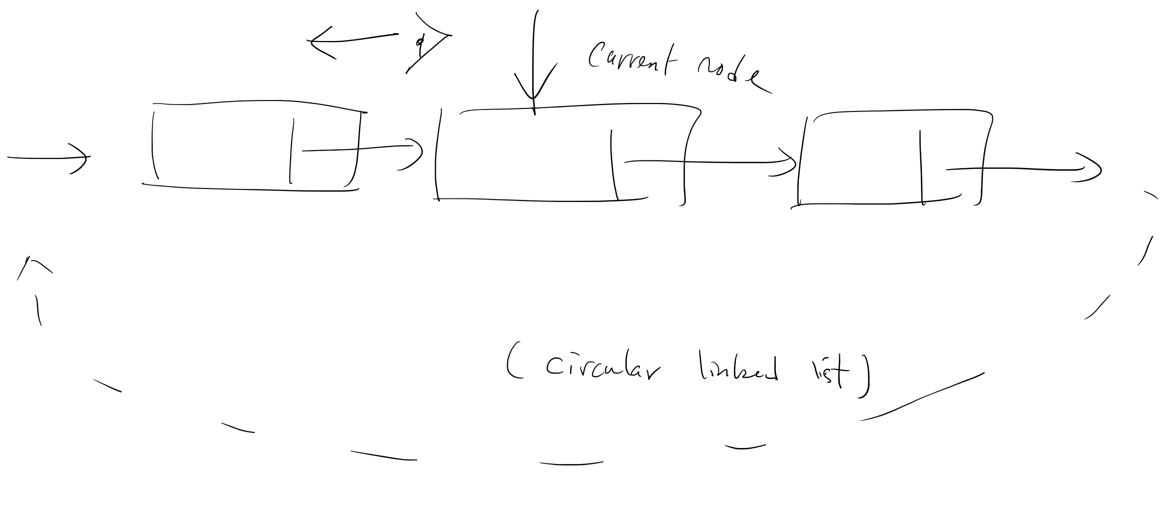
main point

pointers are addresses

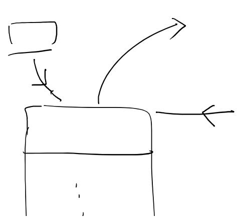
---

difference : singly linked list AND doubly linked list



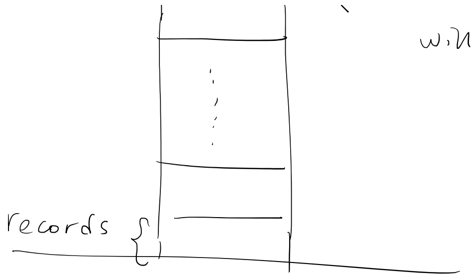


Stacks



All operations will take place here

Top



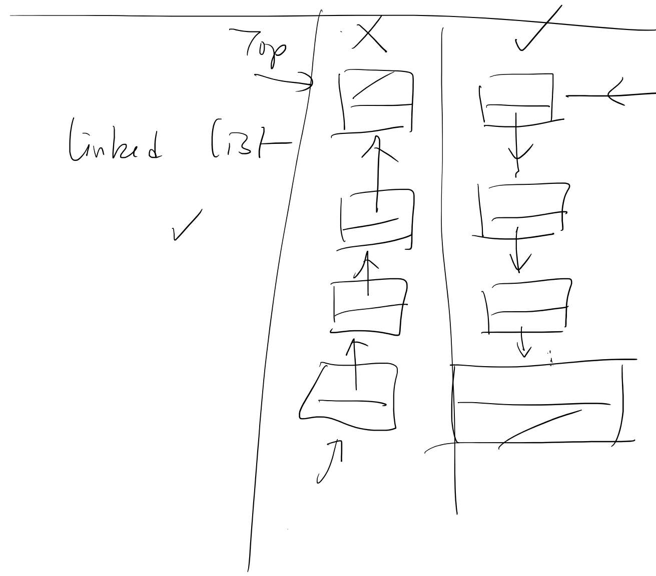
will Take place here

Top

push

pop

Array ✓

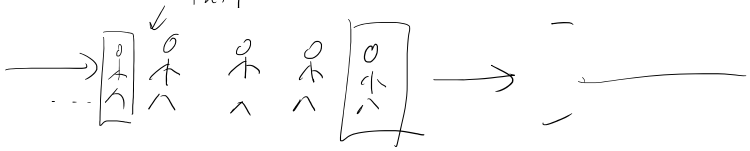


Linked list ✓

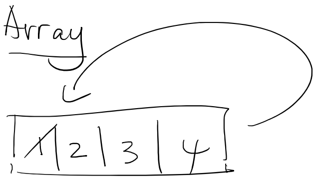
push/pop

(~ Inserting/removing a new obj at the head of a list)

Queue.



Array



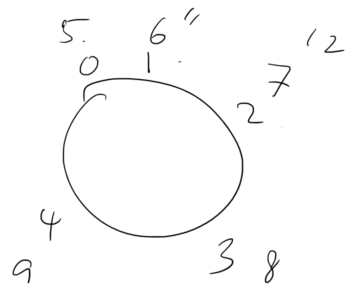
Linked list

enqueue 1
" 2
" 3
deq 1
enqueue 4

how to Specify /determined if a queue is full?

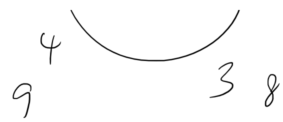
Modulo function

10 mod 5



deq 1
enqueue 4
enqueue 5

if a queue
is full ?
(empty)



9-12-2016

Monday, September 12, 2016 6:47 PM

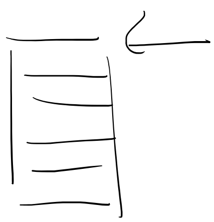
Elementary Data Structures ..

	Stacks	Queues
Sequential Allocation	✓	✓
linked Allocation	✓	All Brass's example

Stacks : LIFO

Queues : FIFO

dequeue

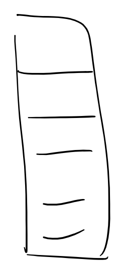


Top

perform all operations at 1 position

Stacks

(dequeue)



front

rear

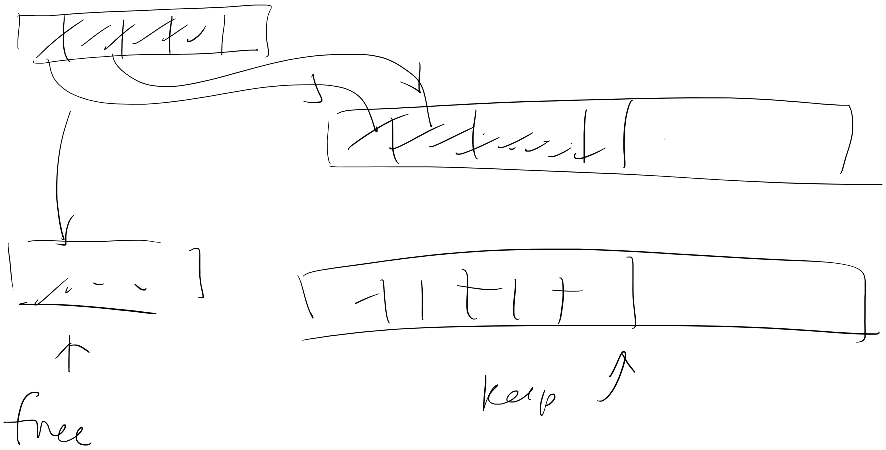
(enqueue) only

Vectors

approaches the limit, double the size of the array

(1)

(2)

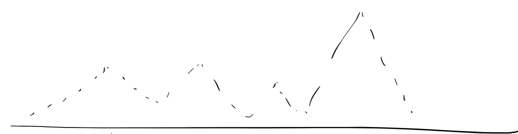


Amortized Analysis

$\boxed{\text{Stacks}}$

↑ push ↑ pop

Running a program



a sequence of DS operation

← n operations →

worst case $T(n)$ time

$$\frac{T(n)}{n}$$

$$T(n) \leq 3n \quad \cdots \quad T(n) \leq c \cdot n$$

↑ constant

$$\therefore \frac{T(n)}{} \approx c$$

$$\frac{T(n)}{n} \approx c$$

constant

$$\frac{T(n)}{n} = \Theta(1)$$

Stacks (CLRS Ch. 17.1)

push pop multipop

$\longleftarrow \quad n \quad \longrightarrow$

d.s. operations

$$\frac{T(n)}{n}$$

more realistic measure.

$$T(n) \le 2n$$

9/13/2016 (CIS 477)

Tuesday, September 13, 2016 9:35 AM

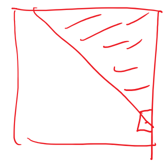
```
for (i = 1, ..., n) {
    for (j = 1, ..., n)      [i+1 ... n]
        [ do ~~something~~
}
```

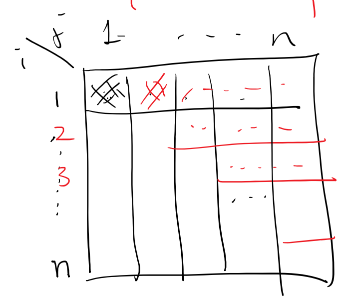
→ constant # of steps

≤ 5

$\Theta(1)$ 1 $\leq 5 n^2$

$1 + 2 + \dots + n-1 = \frac{n(n-1)}{2}$ Work $\leq 5 \frac{n(n-1)}{2}$





$\Theta(n^2)$

for (i=1 ... n) {

$\left[\sum_{j=i+1}^{n} 1 \right]$

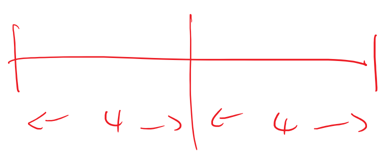
}

$\sum_{i=1}^{n} \left(\sum_{j=i+1}^{n} 1 \right)$

HW. $\sum \sum \sum$ O

Merge Sort: Divide & Conquer. & "Merge"

$T(8) = T(4)$
$+ T(4)$
$+$ merge step $= 2\,T(4) +$ merge step



Recurrence tree (in the slide p. 7)

$T(2000)$

$= 3\,T\left(\frac{2000}{4}\right)$

$T(n) = cn^2 + T\left(\frac{n}{4}\right)$

$+ T\left(\frac{n}{4}\right)$

T $n\Theta(1)$ n ~~3~~

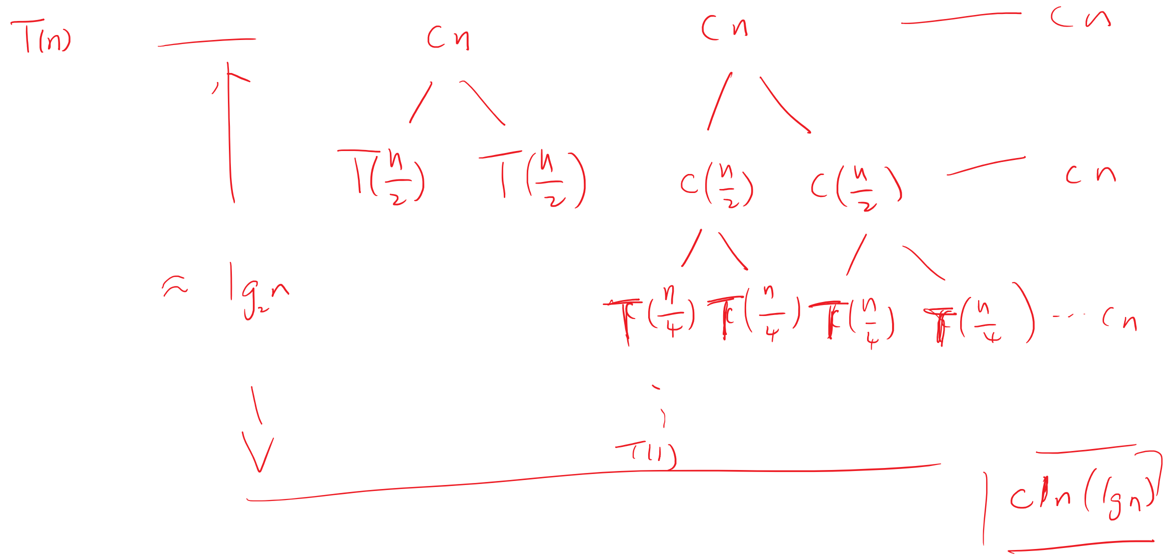
$= 3T\left(\frac{2000}{4}\right) + c(2000)^2$

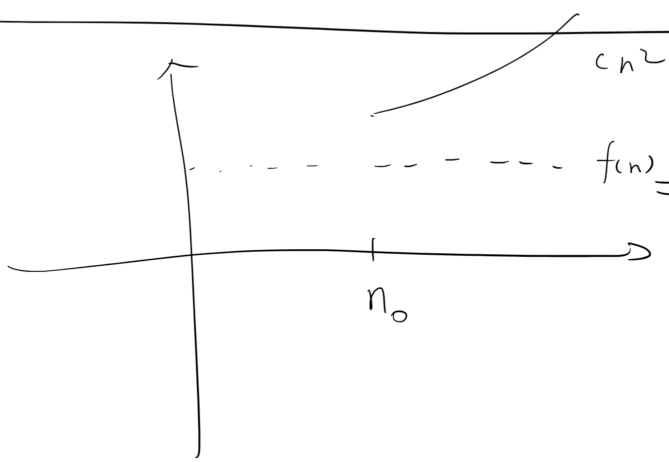
$= 3\left[3T\left(\frac{2000}{16}\right) + c\left(\frac{2000}{4}\right)^2\right] + c(2000)^2$

$$\boxed{T(n) = \begin{cases} \Theta(1) & n \le 1 \\ 3T\left(\frac{n}{4}\right) + cn^2 & \text{o.w.} \end{cases}} + T\left(\frac{n}{4}\right) + T\left(\frac{n}{4}\right)$$

Recursion Tree for Merge Sort

$$\boxed{T(n) = 2T\left(\frac{n}{2}\right) + \cancel{\Theta(n)}\, cn}$$



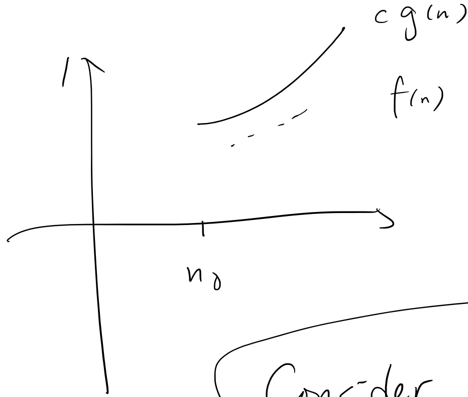


$f(n) = O(g(n))$

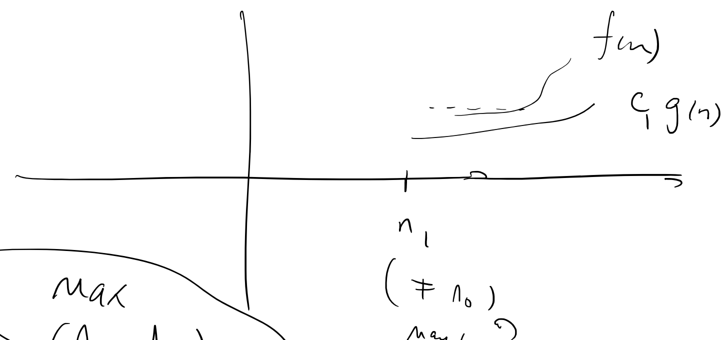
$g(n) = \Omega(f(n))$

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

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



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

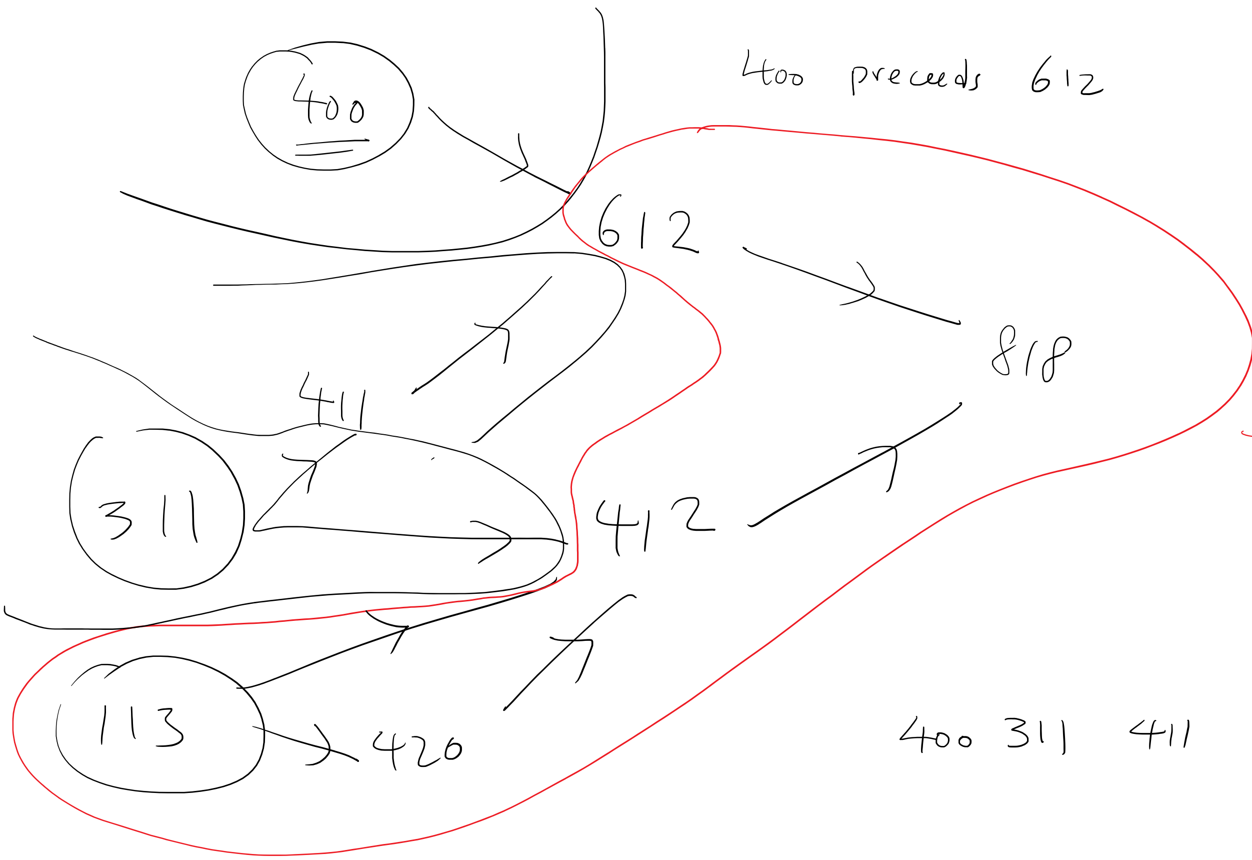


Consider $n \geq \max(n_0, n_1)$

9-14-2016

Wednesday, September 14, 2016 6:43 PM

400 precedes 612



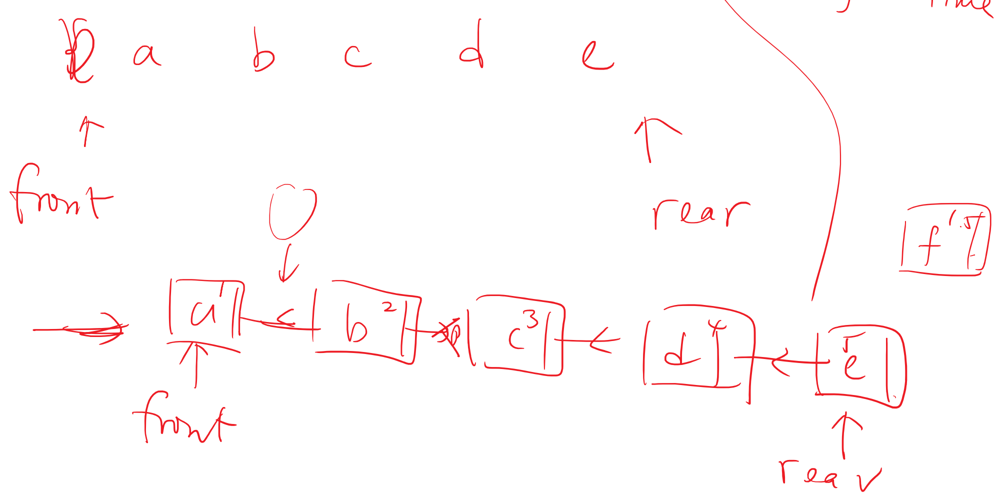
topological

400 311 411

priority | Queue

FIFO

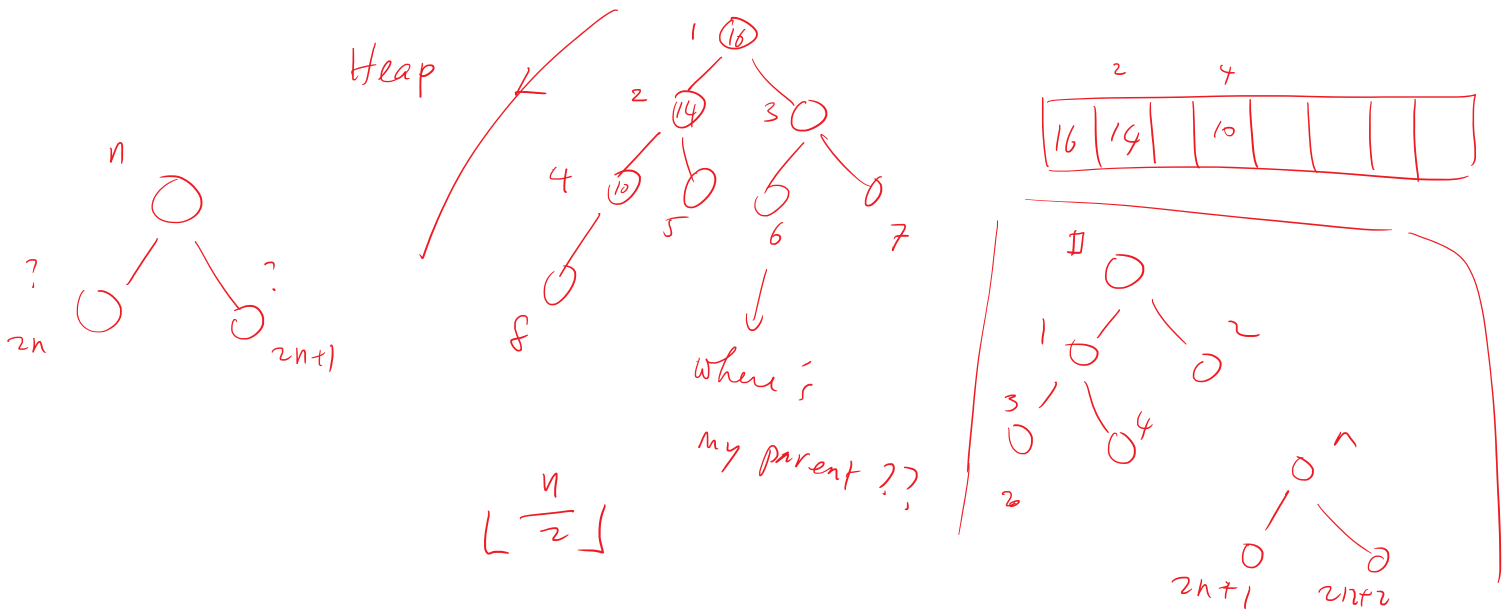
priority : time



Priority Queue

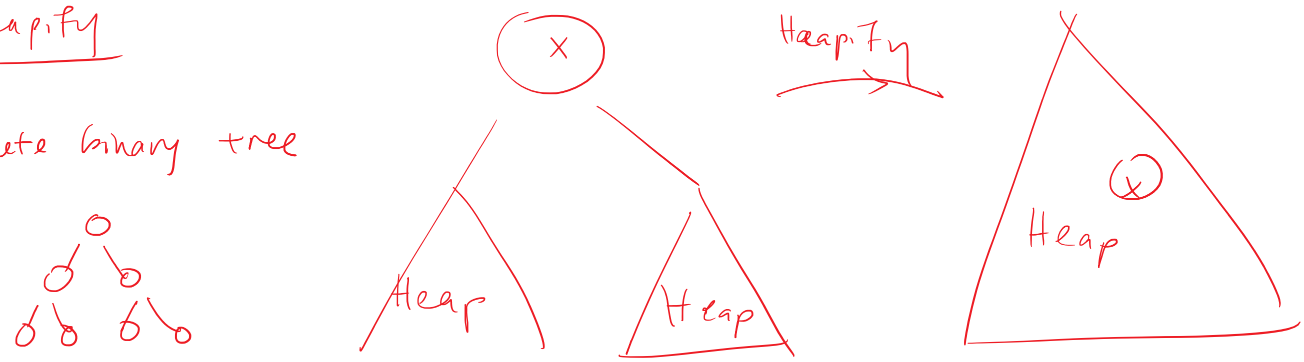
↳ may use "Heap" to implement

Heap

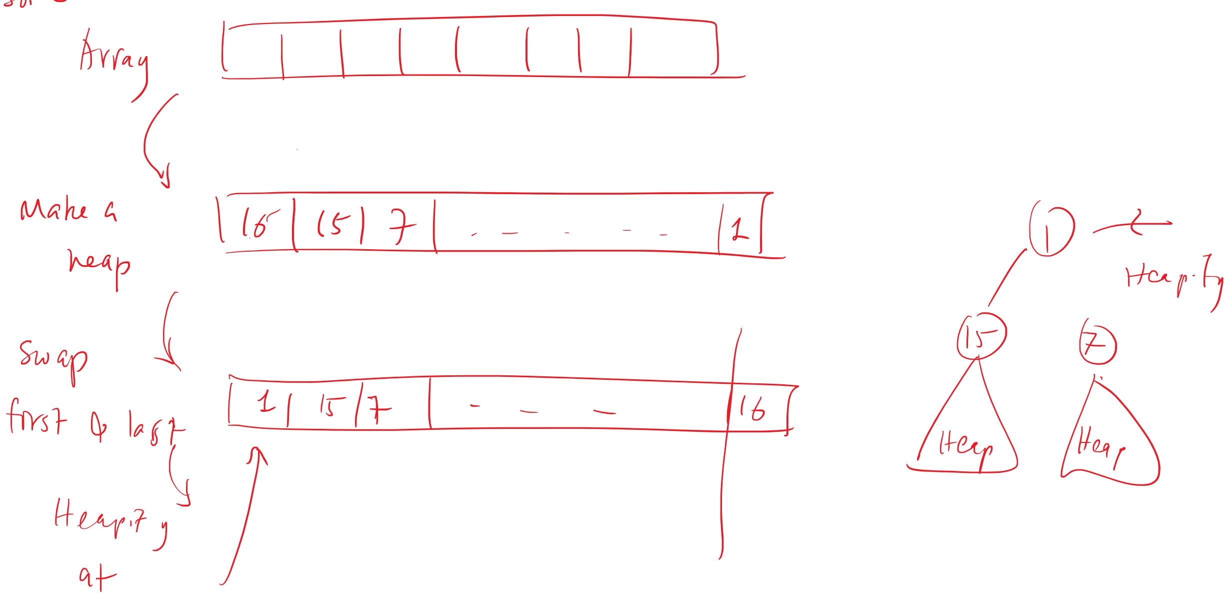


Heapify

Complete binary tree



Heap Sort



↳ Complete binary tree of n nodes

height $= \sim \lg_2 n$?

$2^h \simeq n$

02-TopologicalSort

Module: Elementary Data Structures
CSE 674 Advanced Data Structures and Algorithms
Class Activity: A data structure for topological sorting

Question

Consider the following table, which record a number of college courses and their pre-requisites.

Course	Pre-requisites
612	400, 411
818	412, 612
411	311
412	113, 311, 420
420	113

Specify a partial order

Design a data structure and use it to compute an order of taking these courses so that the pre-requisite requirements are all met.

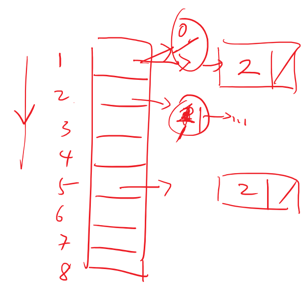
Answer

1. Simplify.

1	2	3	4	5	6	7	8
410	612	818	412	411	420	311	113

2. 612 requires 400
612 " 411

$400 \rightarrow 612$
$411 \nearrow$

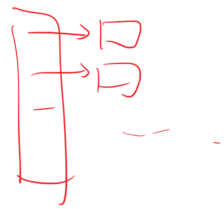


1 7 5 2 8 6 4 3

Answer: .

m succ.-pred relations

O(m)

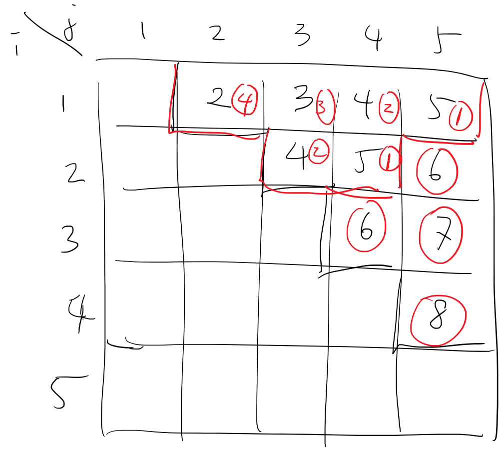


9-15-2016

Thursday, September 15, 2016 9:38 AM

HW 1 :

Q1



for (i = 1 ... n)

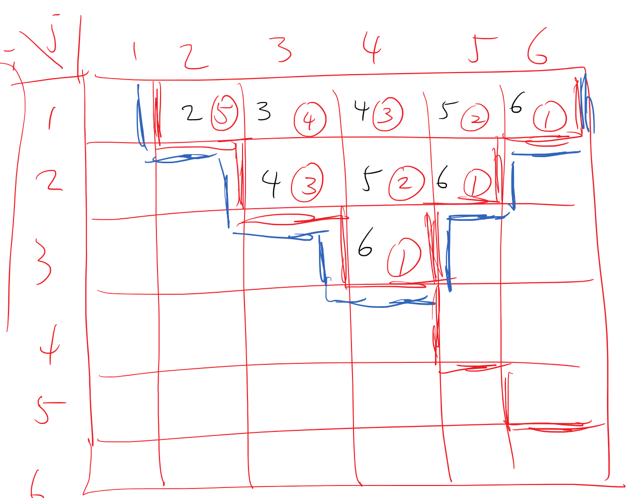
for (j = i+1, ... n)

{

}

n = 5

i + j - 1



$1 + 2 + \cdots + 2k-1$

$R_k \quad \frac{2k(2k-1)}{2}$

$R_1 + \cdots + R_k$

$R_k = 1 + 2 \cdots + \cdots + 2k-1$

$R_2 = 1+2+3$

$R_1 + \cdots + R_k$

n = 6

1 2 3 4 5

1 2 3

1

$n = 2k$ $\quad k = n/2$

k

[1 2 3 4 ... 2k-1]

1 2 3 4 ... 2k-3

1 2 3 4 .. 2k-5

1

$$Ans = R_1 + \cdots + R_k$$

$$R_k = k(2k-1)$$

$$Ans : \sum_{i=1}^{k} i(2i-1) = 2\sum_{i=1}^{k} i^2 - \sum_{i=1}^{k} i$$

Recurrence

Recurrence

(1) Substitution Method [- -]

(2) Recursion Tree [Guess]

(3) Master Thm [Apply Check]

Seldom in this class

Example for Merge ~~Sort~~ $f(n) = \Theta(n)$ $a = b = 2$

Master Thm, which case matches ?

By Master Thm

$\log_b a = 1$.

$f(n) = \Theta(n^{\log_b a})$

$\therefore T(n) = \Theta(n \lg n)$

$f(n) = 1$

$a, b = 2$.

$T(n) = 2T\left(\frac{n}{2}\right) + 1$ Case 1

$T(n) = \Theta(n)$

9-19-2016

Monday, September 19, 2016 6:52 PM

HW 1. linked allocation

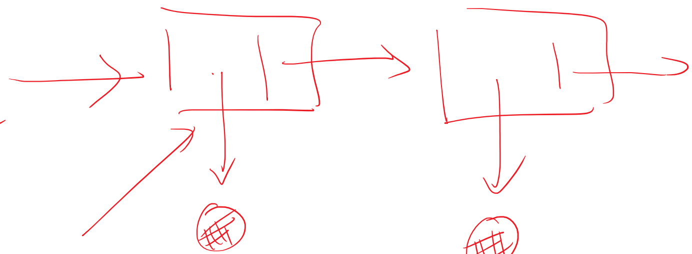
Heap "Structure"

Sorting

Priority Queue

HW 1 :-

mylist

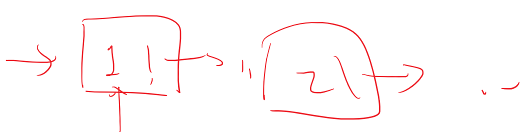


Generic

External linked list

Internal linked list

* requirement :-



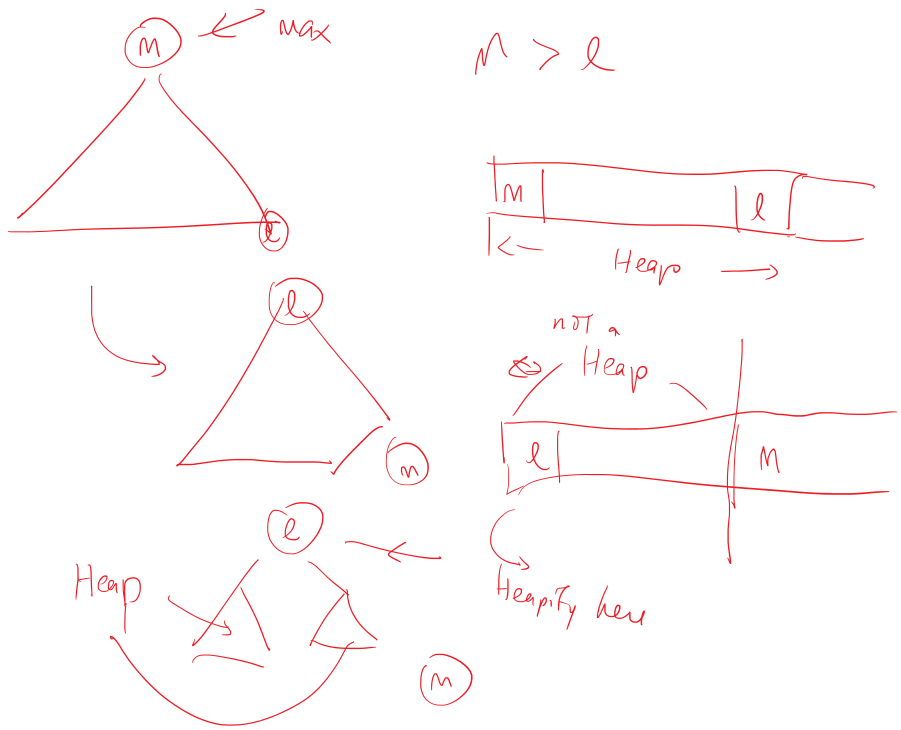
HeapSort (A[1, ..., n])

O(n) 1 Build a Heap

Time

(Amor Analysis)

~ n lg n



HeapSort

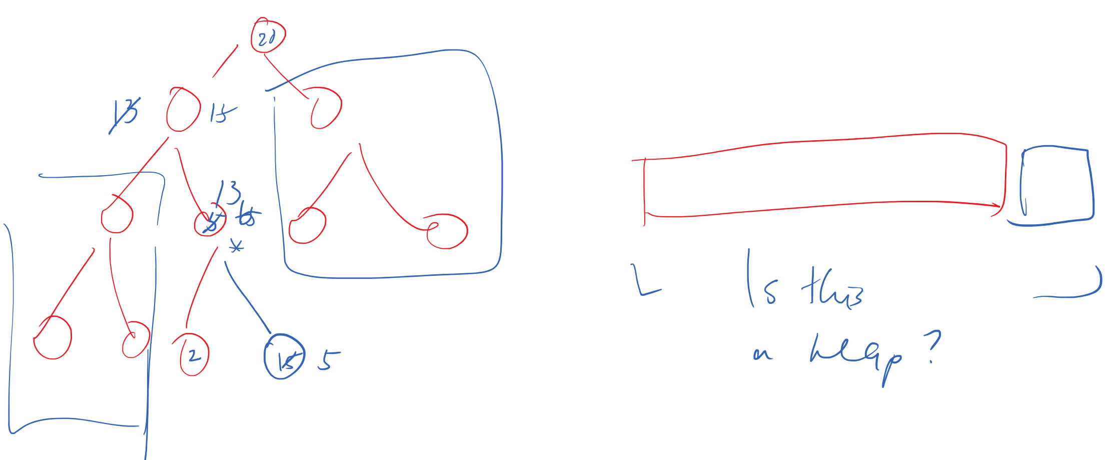
is O(n lg n)

Time.

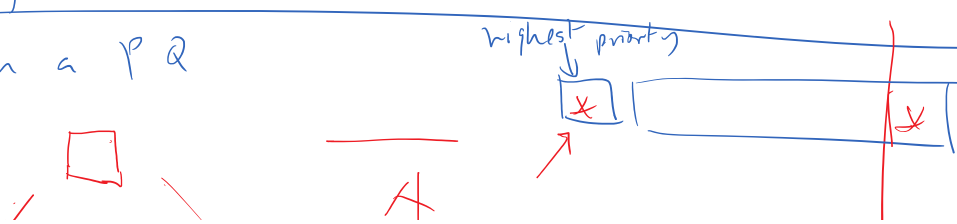
A.D.T Queue Enqueue + dequeue remove

Priority Queue ✓ ✓

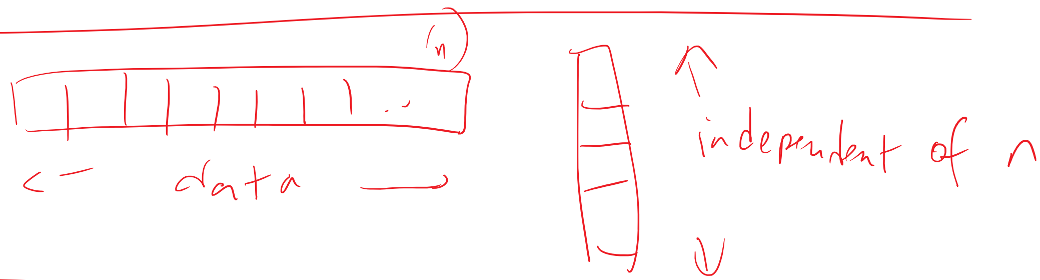
Insert



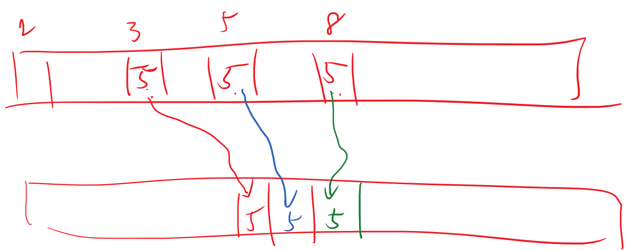
dequeue from a PQ



in-place sorting



Stable sorting



9-20 / CIS 477

Tuesday, September 20, 2016 9:24 AM

2016

January

S	M	T	W	Th	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

February

S	M	T	W	Th	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29					

March

S	M	T	W	Th	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

April

S	M	T	W	Th	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

May

S	M	T	W	Th	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

June

S	M	T	W	Th	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

July

S	M	T	W	Th	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

August

S	M	T	W	Th	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

September

S	M	T	W	Th	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

October

S	M	T	W	Th	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

November

S	M	T	W	Th	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
~~20~~	~~21~~	~~22~~	~~23~~	~~24~~	~~25~~	~~26~~
~~27~~	28	29	30			

December

S	M	T	W	Th	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Tentative dates

EXAM 3: Dec 1, Dec ~~8~~6

Exam 2: Nov 3

EXAM 1: Oct 6, Oct 11

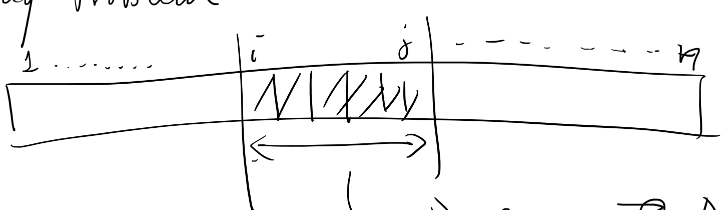
Today Topic: ——

Divide & Conquer Paradigm.

Recall: "Recurrence" + "Master Thm"

Analysis of Alg using

Max. Subarray Problem

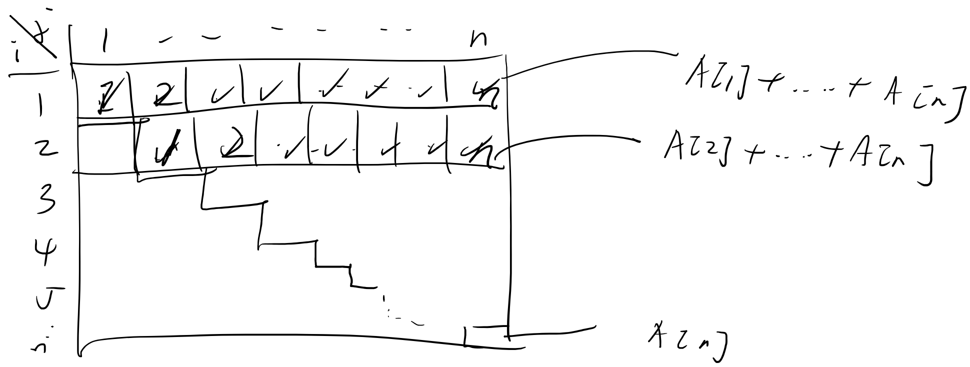


$i \le j$

sum $= A[i] + \cdots + A[j]$

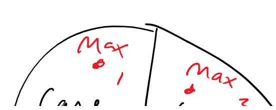
Brute Force ——

$$\sum_{j=1}^{n} \frac{n(n+1)}{2}$$

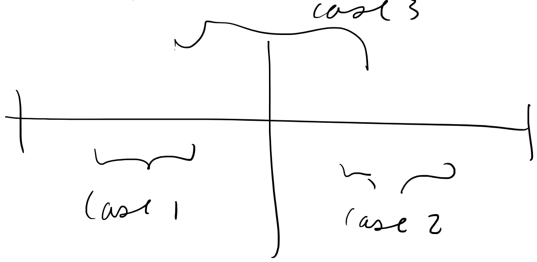


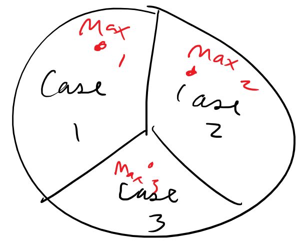
Divide & Conquer

case 3



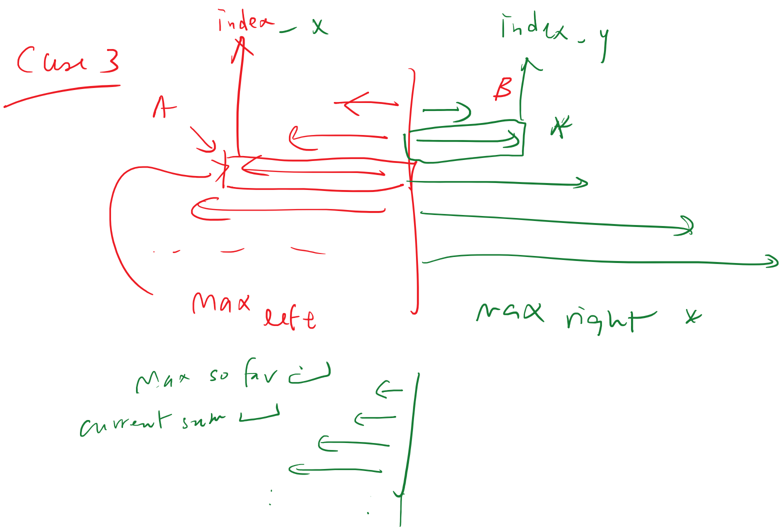
Divide & Conquer

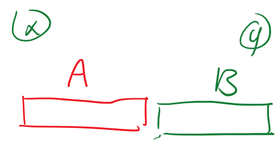




Max = Maximum of
Max 1, Max 2
Max 3

Case 3





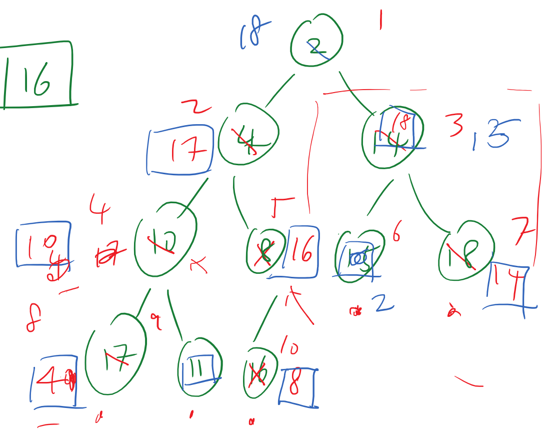
9/21/2016

Wednesday, September 21, 2016 6:48 PM

[2, 4, 14, 10, 8, 15, 18, 17, 11, 16]

| 2 | 4 | 14 | 10 | 8 | 15 | 18 | 17 | 11 | 16 |

Max Heap? →



Step 1: Build Heap

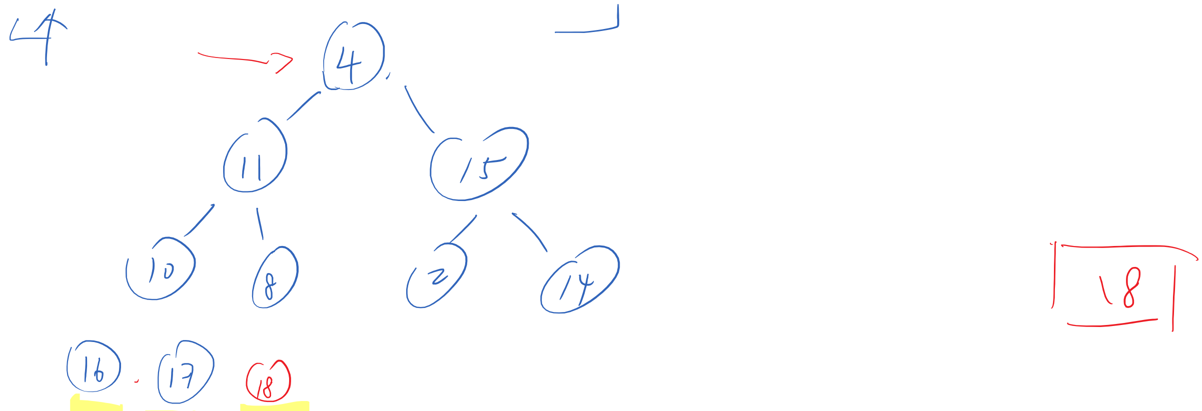
$\lfloor \frac{n}{2} \rfloor$

| 18 | 17 | 15 | 10 | 16 | 2 | 14 | 4 | 11 | 8 |

Step 2

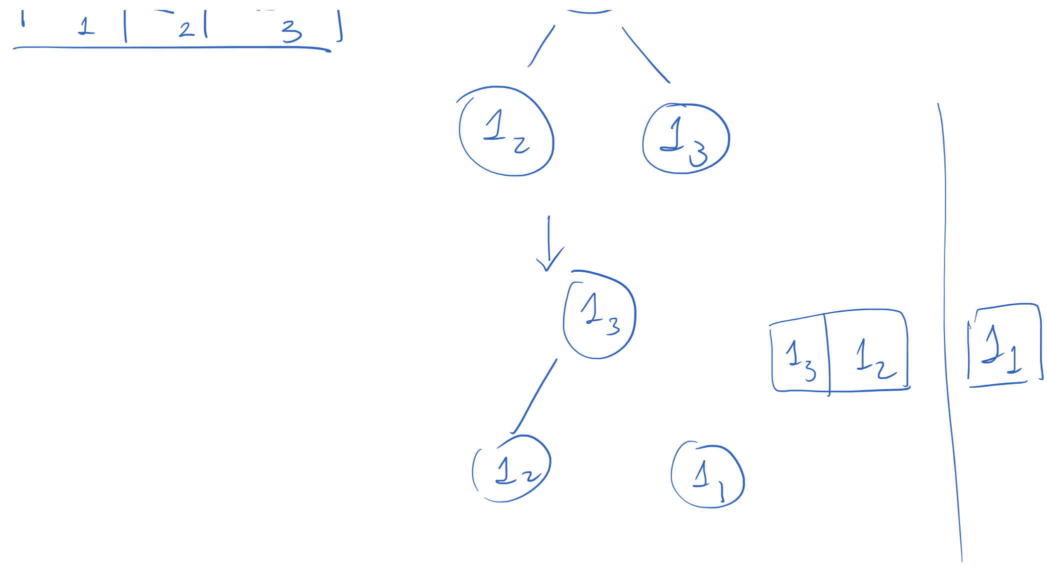
| 8 | 17 | 15 | 10 | 16 | 2 | 14 | 4 | 11 | 18 |

4

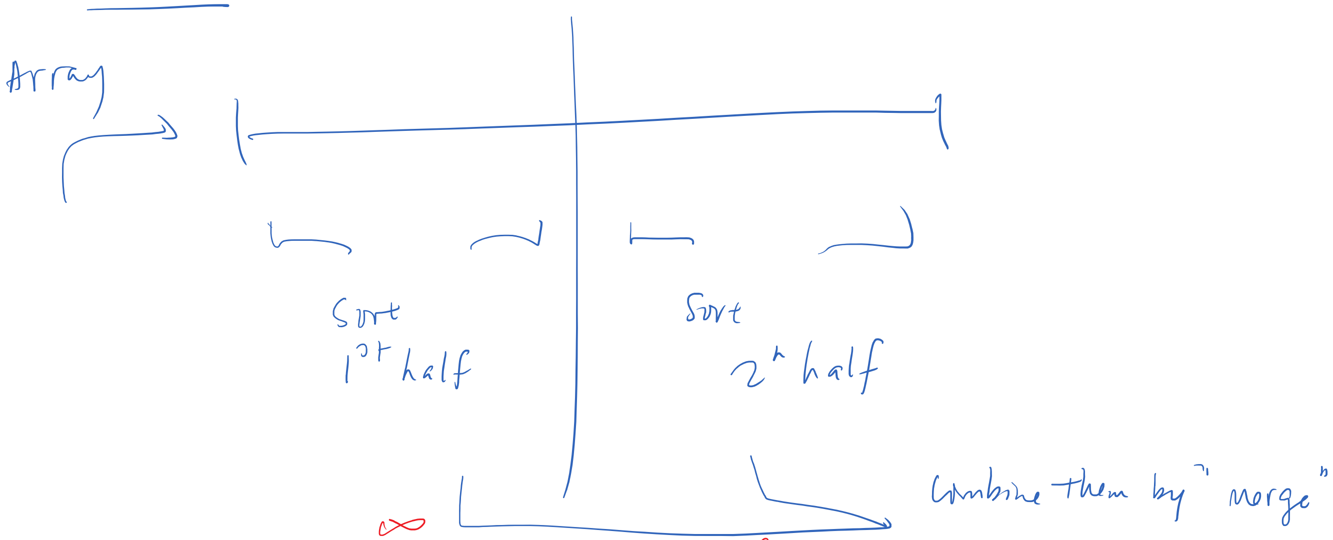


Heap Sort is not stable

| 1_1 | 1_2 | 1_3 |



Merge Sort (divide & conquer)

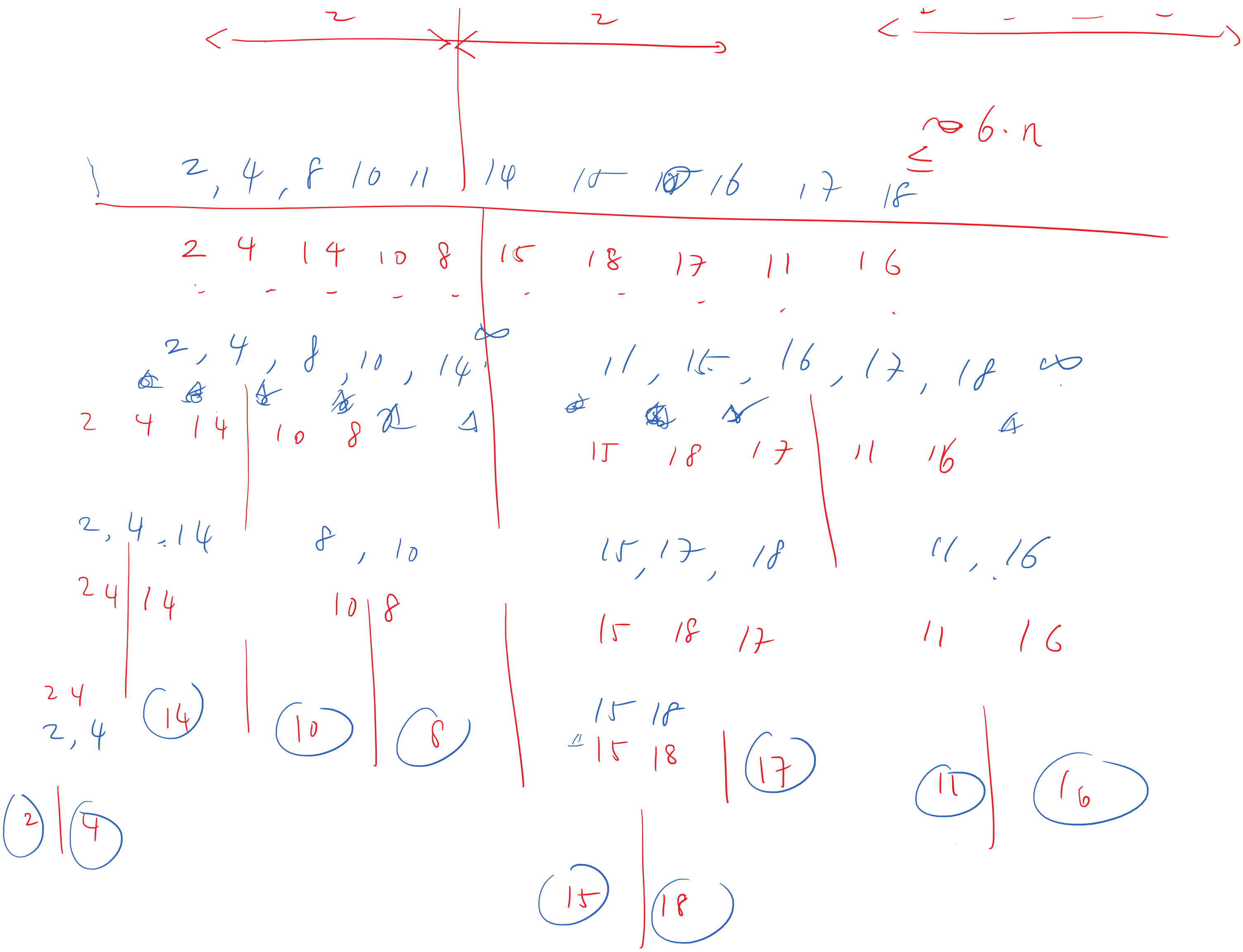


1 3 7 8 | 6 9 10 11

sorted seq | sorted seq

1 3 6 7 8 9 10 11

$\frac{n}{2}$ | $\frac{n}{2}$



for divide & conquer alg:

$$\frac{n}{b} \quad \frac{n}{b} \quad \cdots \quad \frac{n}{b}$$

b parts

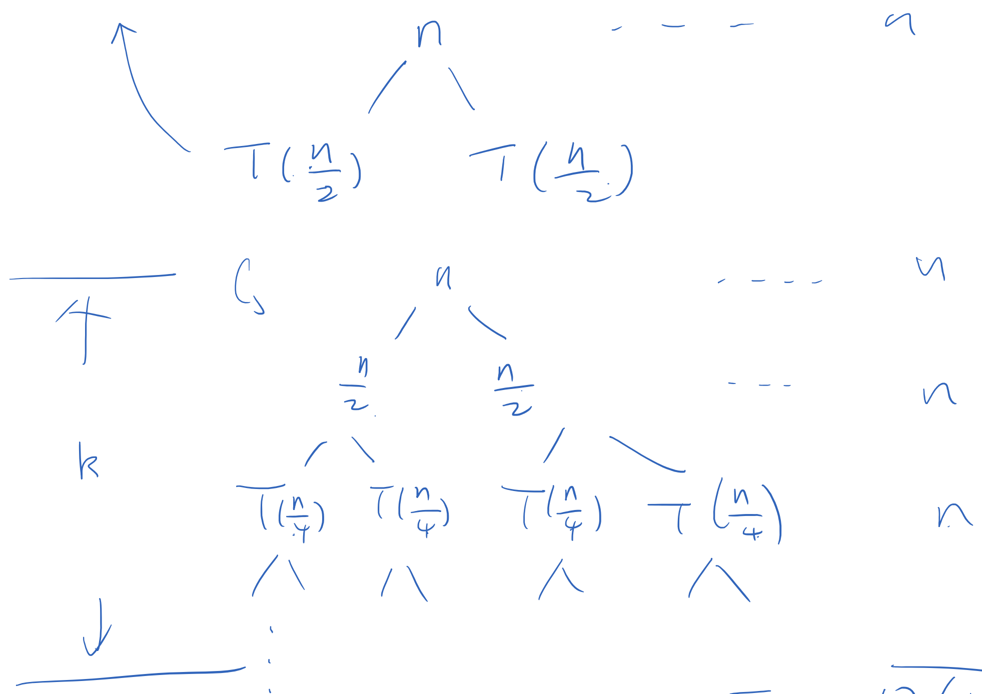
b subproblems

Apply the method to a of them

$$\underbrace{T\left(\frac{n}{b}\right) + \cdots + T\left(\frac{n}{b}\right) + T\left(\frac{n}{b}\right)}_{a \text{ subproblems}}$$

$$T(n) = \begin{cases} 1 & n \le 1 \\ 2T\left(\frac{n}{2}\right) + n & n > 1 \end{cases}$$

$T(n)$

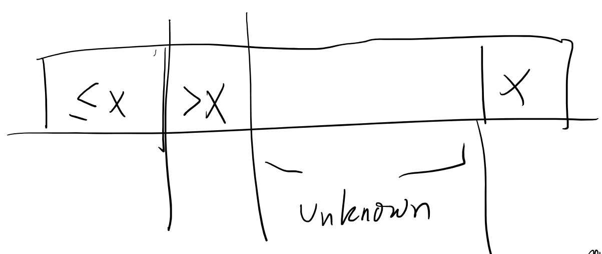


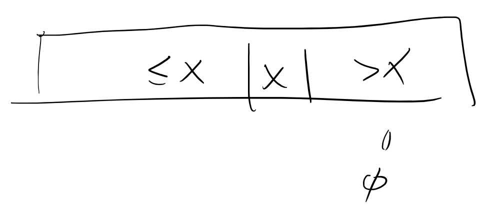
$2^k \simeq n$

$k \simeq \lg n$

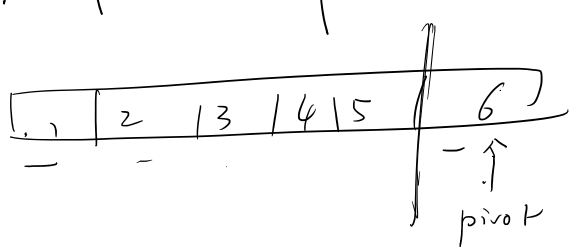
$T(n) = \Theta(n \lg n)$

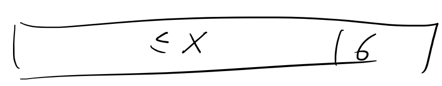
Quick Sort :





e.xamp





1. Hand in HW 2. (Now.) to Sid
2. "Job Fair"
3. "Orange Success"
4. "Divide & Conquer"

Example: QuickSort

$$T(n) = T(n-1) + \Theta(n)$$

$$T(n) = T(n-1) + cn$$

$$= T(n-2) + c'(n-1) + c(n)$$

$$= T(n-3) + c''(n-2) + c'(n-1) + cn$$

$$\therefore T(n) = \cdots = \Theta(n^2)$$

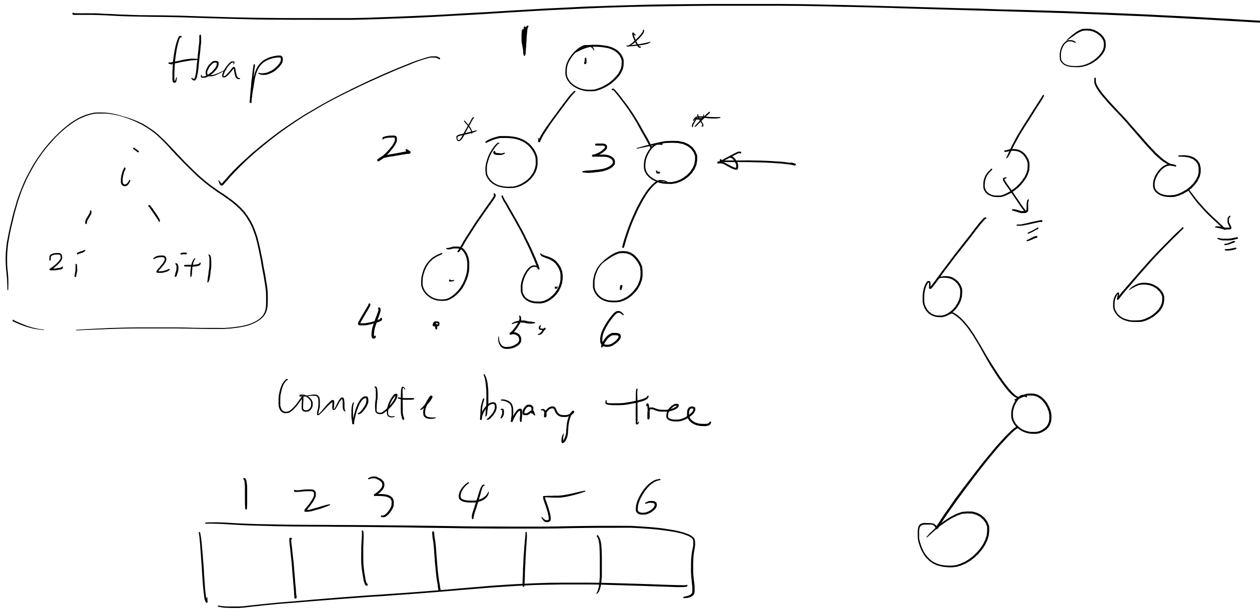
(worst case)

Quicksort worst case: $\Theta(n^2)$

→ Average case: $\Theta(n \lg n)$ [requires new techniques]

Heap

Heap

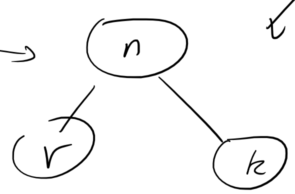


Complete binary tree

Heap

satisfy the max Heap property ✓

for any non-leaf node

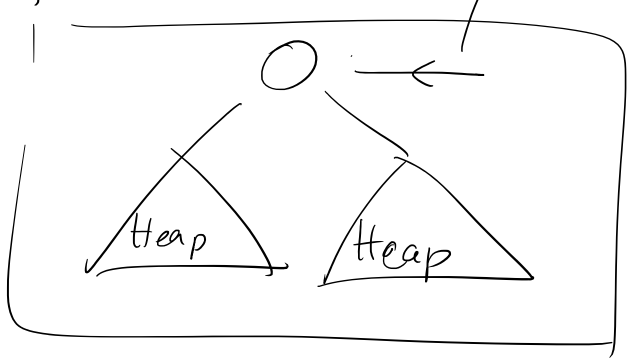


$n \geq r$

$n \geq k$

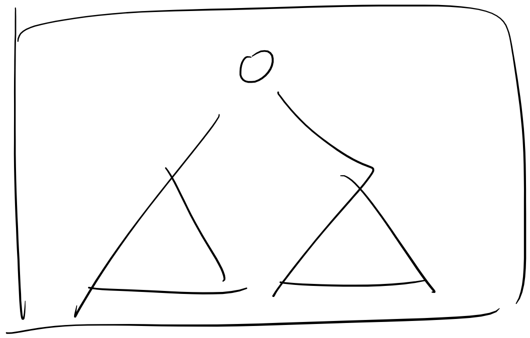
To build a heap, we need

heapify



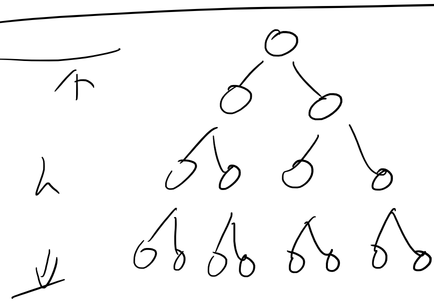
not a heap

After applying heapify.



Heap

About complete binary tree



a complete binary tree of height h has at most

$$1 + 2 + 2^2 + 2^3 + \cdots + 2^h \text{ nodes} = 2^{h+1} - 1$$

$$1 + r + r^2 + \cdots + r^h = \underline{r^{h+1} - 1}$$

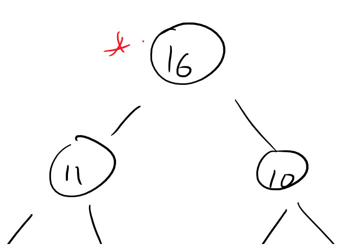
$$2^h \simeq n \text{ nodes}$$

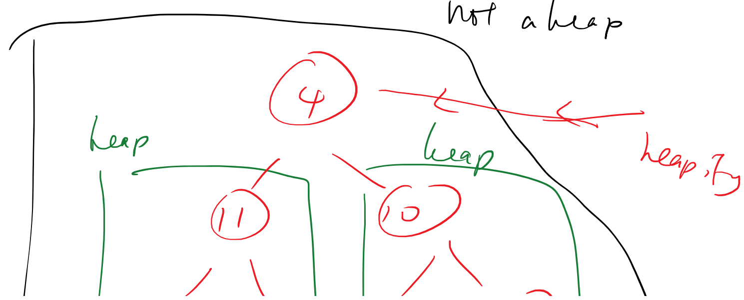
$$h \simeq \lg_2 n$$

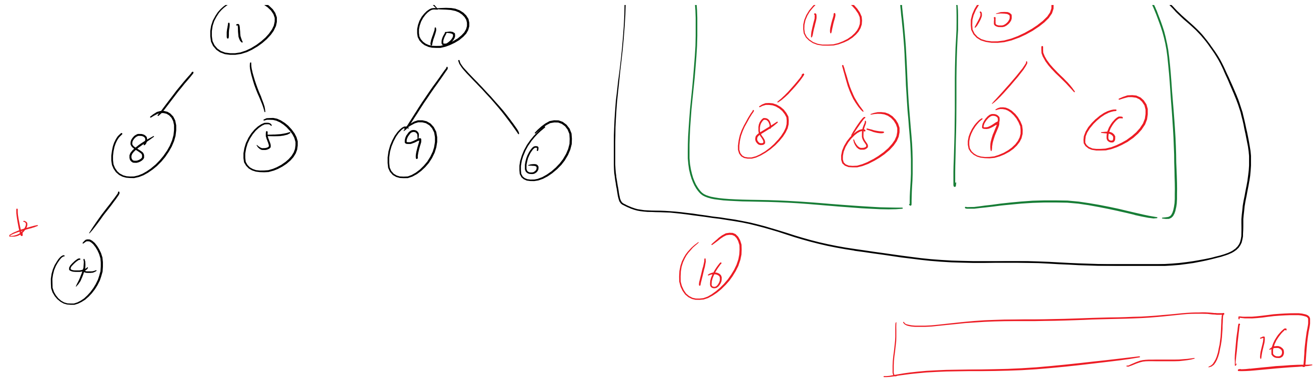
Heapsort

1. Build a heap.
2. "Keep heapify" at the first element

$r \geq 1$






11
8
5
4
10
9
6
11
8
5
10
9
6
16
16

9-26-2016

Monday, September 26, 2016 6:47 PM

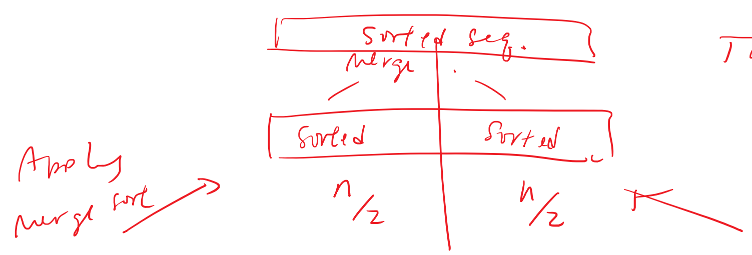
Midterm:

** divide & Conquer paradigm :— **

Analysis Time Complexity ⟨ recursion tree to guess; Master Thm to verify.

Correctness { use strong induction as a guideline

Merge Sort: $P(n)$: Merge sort is correct when The input size is n.



$P(n)$: $P(\lfloor \frac{n}{2} \rfloor)$ $P(\lceil \frac{n}{2} \rceil)$

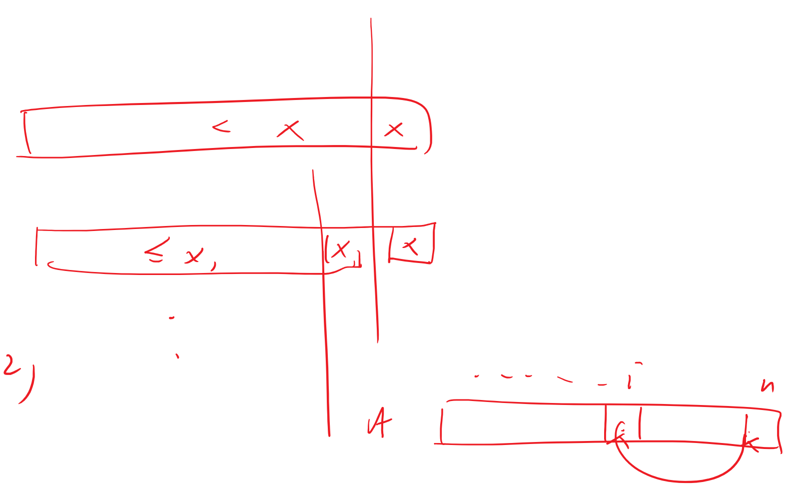
Smaller case

Quicksort :—

Worst Case

$T(n) = T(n-1) + \Theta(n)$

$\hookrightarrow$ $T(n) = \Theta(n^2)$



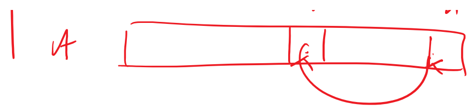
To

/ strategy

To improve

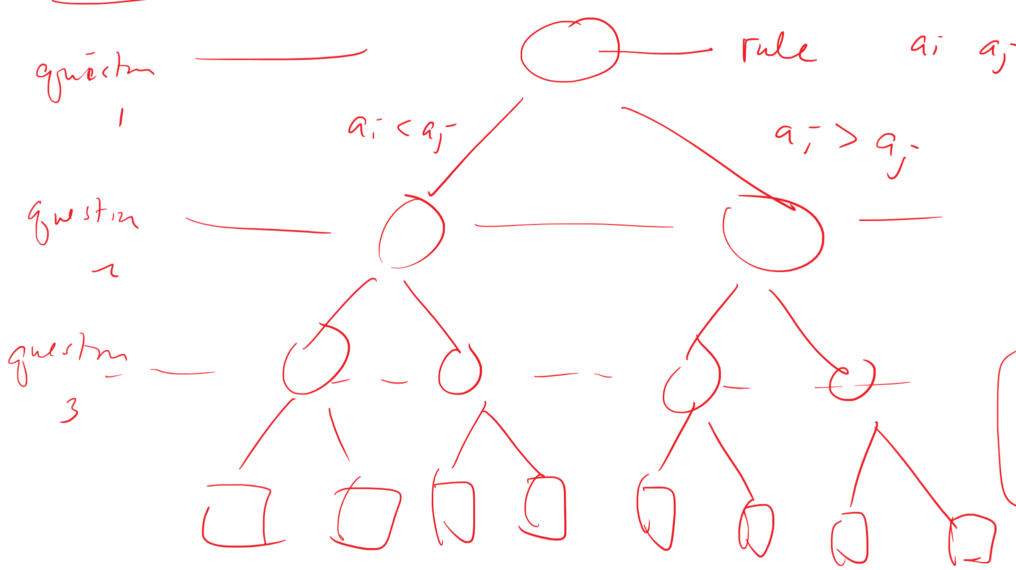
strategy.

- → select a random ~~element~~ position i
- → Swap the element at $A[i]$ to $A[n]$



↳ Average case $\approx \Theta(n \lg n)$

Comparison Based Sorting method.. —



n distinct elements

n! different possible permutations.

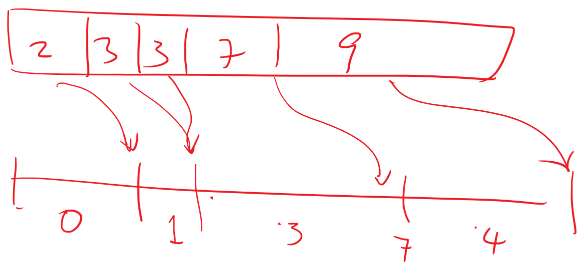
To distinguish ← cases

you need at least $\Theta(n \lg n)$ questions

Counting Sort

- Stable
- not comparison based

0	1	2	3	4
2	1	0	4	2



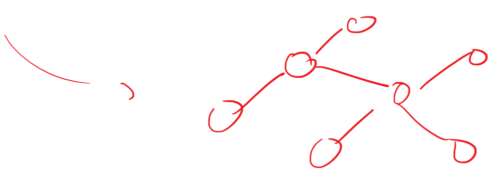
Pseudocode (given in the slide)

Analysis $\Theta(n) + \Theta(k)$

$\Theta(n+k)$

Leonardo numbers → Smoothsort (next time)

Trees → (HeapSort)



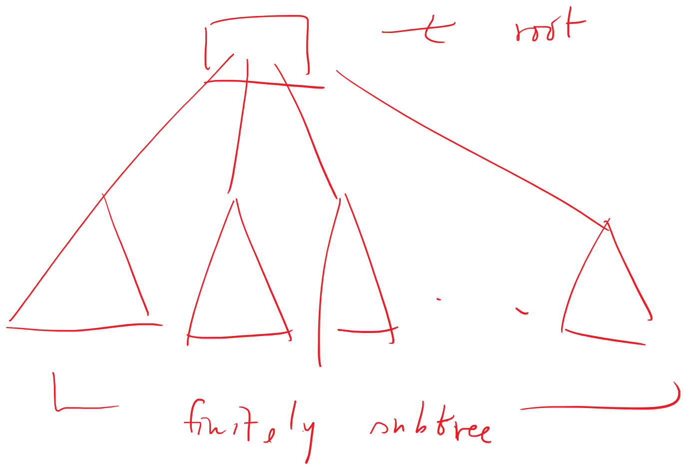
Tree Trees

n vertices

n-1 edges

connected (Graph)

Rooted Tree



Base Case o root
a tree

General Case