Cheat Sheet ryan wentwo

Postfix x+y-z+t → xy+z-~~t~~+

Purpose of Stack in postfix → push ops before enqueing them

$T(n) = T(n-1)+1$ → $O(n)$

Tree h=4 → nodes max is 15

quicksort same best case as merge and worst case of insertion

Comp for unbal BST → $O(n^2)$

find in bal BST → $O(\log_2(n))$

Display(print) BST in sorted order → $O(n)$

insert in Bal BST → $O(\log_2(n))$

Every AVL is BST → true

AVL makes BST of $\log_2 n$ → True

Comparisons AVL tree → $O(n \log_2 n)$

Every AVL Tree is R-B Tree → True

AVL tree is more compact then R-B

2-4 tree largest when → 2 node

2-4 tree smallest when → 4 node

external Child for 2 node → 3 child

2-4 trees → All ext nodes have same depth

↳ apply inorder / ext node differs by 1 from int nodes

insert op overflow if → 4 node

num comp for unsuccessful 2-4 → $O(\log_2 n)$

Skip list are non-deterministic

queue not part of dictionary

insert function is random → Skip lists

not unsuccesful search $O(\log_2 n)$ → Skiplists

Comp for delete in Skip → $O(\log_2 n)$

find single vertex indegree ~~[illegible]~~ wit n vertices → $O(n)$

find single vertex indegree with n and m → $O(n+m)$

bellman-ford → can have negative, Dijkstra no negative values

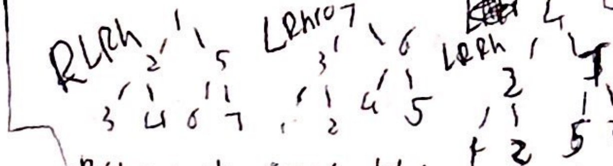
remove max/min is binary heap worst → $O(\log_2 n)$

Huffman Algo for file compression → output binary tree bottomup

↳ ext nodes contain text file Characters / have info about frequencies of Characters in a text file.

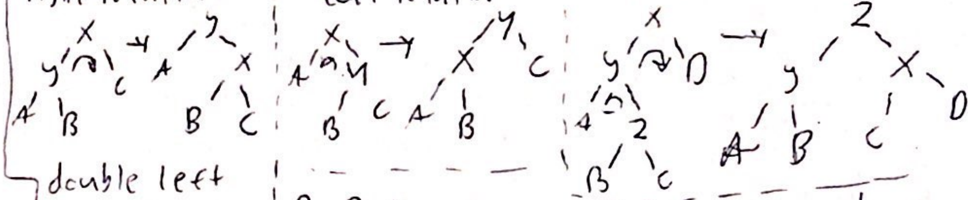
insert into MPQ → $O(n)$

Trees: Preorder Postorder Inorder

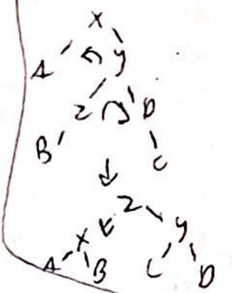


- BST insert, search, delete is all ~~[illegible]~~ $O(\log n)$ stairs sol
- unbalance one sided have worst case scenerio

- AVL tree has balance factor -1, 0, 1. All AVL are R-B tree
- AVL Tree height is $h < 1.44 \log_2(n+1) - 1.328$ n = nodes
- right rotation | left rotation | double right



double left



- R-B tree: root is black and red.
- All kids black + external, no two ~~back~~ back red
- $H = O(\log_2 n)$ for R-B. Insert $O(\log_2 n)$
- 2-4 - Each node has least 2 vals
- $h \leq \log(n+1)$. Search $O(\log n)$
- overflow if 4 node bc more then 4 kids → to solve split and push new val to node space above and create new child
- when Delete replace by inorder successor

- underflow when delating makes a 1 node, either combine nodes or transfer values.

	find	put	erase	
hash	1	1	1	→ simple to make
Skip	Logn	→		→ simple
AVL 2-4	Logn	→		→ complex

Hash table $h(k) = k \bmod m$

Collisions: Linear / quadratic ↗ makes it clustered.

Double hash / Chaining

↳ $[h_1(k) + i\, h_2(k)] \bmod m$

Chaining use linked list.

(h_2 = prime - (k mod prime))

Master theorem $T(n) = aT(n/b) + f(n)$

1. $a < b^k$ $T(n) \sim n^k$
2. $a = b^k$ $T(n) \sim n^k \log_b n$
3. $a > b^k$ $T(n) \sim n^{\log_b a}$

$f(n) = cn^k$

Stack → push front pop end / queue → push back pop back

Tail recursion use of only one recursive call at end of function

merge Sort $O(n \log_2 n)$ comparisons where n is length

Mergesort run time is $T(n) = 2T(n/2) + \Theta n$.

QuickSort run times: worst → $T(n-1) + \Theta n$ / $O(n^2)$

Best case → $T(n) = 2T(n/2) + \Theta(n)$ / $O(n \log_2 n)$

Avg case → $T(n) = T(n/4) + T(3n/4) + \Theta(n)$ / $O(n \log_2 n)$ → specific case

worst case is sorted for quicksort and pivot effects run time.

Merge sort not in place Algo.

Priority queue - Prioritized entries of (key, value). total order relation 1. reflexive $x \le x$
2. Antisym $x \le y \to y \le x$
3. Transistive $x \le y \to y \le z \to x \le z$

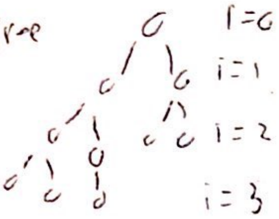
Unsorted List: Insert $O(1)$, remove $O(n)$ | Sorted List insert $O(n)$ remove $O(1)$

Binary heap Insert, remove take $O(\log_2 n)$. $2k+1$ get left child first val pos $= 0$

Min heap is a dense binary tree, $h = \lfloor \log_2 n \rfloor$ $2k+2$ get right child insert/delete $(\log_2 n)$

Heap sort $O(n\log_2 n)$ → avoids overhead association with recursion.

Dense Tree



height m=3

Huffman Algo → technique for data compression
Makes a frequency table with number occurences of given character.

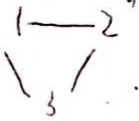
Huffman tree is built in $n-1$ steps merging subtress in bottom up.
~~Runtime. Create O(m)~~

runtime: Create table is $O(m)$, Create MPQ is $O(n)$, extract first and 2nd min and insert $O(\log_2 n)$
→ previous step is $n-1$ times total of $(n\log_2 n)$ so total runtime is $O(m + n\log_2 n)$

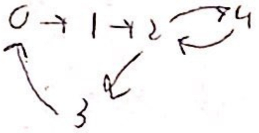
Graphs - pair of vertires and edges. Connected if any 2 verticies there is a path.
acyclic graph is a graph without cycles. Can make Adj matrix of a graph

Directed graph

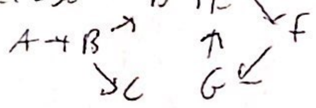
undirected graph



1. Cyclce directed



2. ~~Cycle~~ Cycle



BFS and DFS search to pass through graph.

find shortetes path Dijkstra Algo → $O(|V|+|E|\lg|V|)$ (no neg allowed) or Bellman-ford (can have neg) $O(|V||E|)$ sparse $O(|V^3|)$ dense

Togographical sort by kahn Algo

Kruskal Algo: go smallest by smallest edge, no cycle → make min span tree.

Prim Algo: Start at a node take smallest path, and so on for found vertices. No cycle.

Kruskal runtime is $O((V+E)\lg|V|)$ and is the same for prim

Make set(x) - Creates a new element whos member is x.

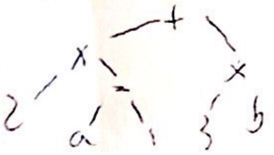
Union(x, y) - set S_x and S_y into $S_x \cup S_y$. | Find set(x) - returns pointer to rep set contain x.

MST is acyclic, connects all Vertices and total weight $= \min \sum w(u,v)$

Infix → postin

$(x+y)^2 + (x-4)/3 \to xy + 2 \wedge x4 - 3/ +$

Binary arithmetic tree of $[2 \times (a-1)] + (3 \times b)$



use specialized inorder traversal to print

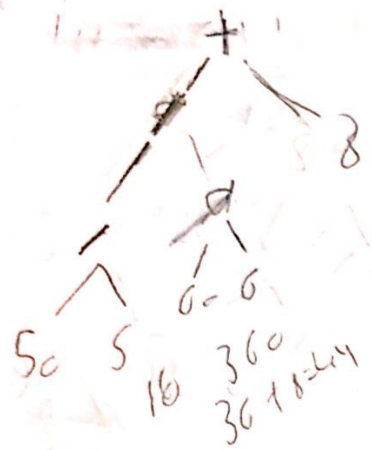
Doublylinked list to make Disjoint.

runtime of Dijikstra as Adj. matrix and mpq is binary heap → $O(n^2\log n)$

Pq selection is $O(n^2)$, insertion $O(n^2)$, heapsort $O(\log_2 n)$

DAG does $O(|V|+|E|)$ time for SSP.

Cheatsheet Ryanwentows



3 4 5 6

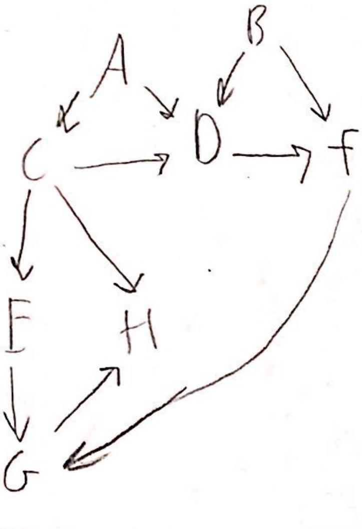
```
  1
 5 0
 1 7 1
+
------
 2 2 1
```

```
   1
 1 2 9
+1 6 7
------
 2 9 6
```

Q35

~~E~~
H
~~G~~
~~D~~
~~F~~
~~C~~
~~A~~

A C D F G H E



Q37 kahns instead.

indeg

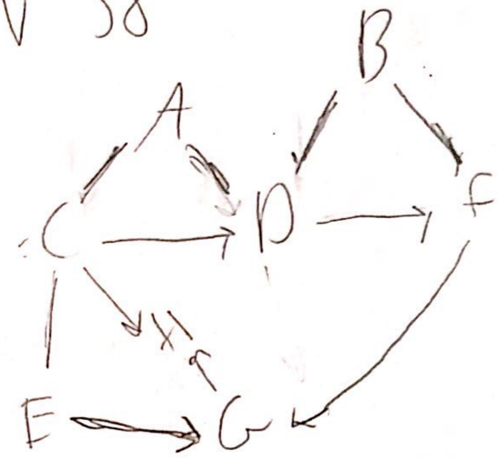
A = 0
B = 0
C = ~~1~~ 0
D = ~~3~~ ~~2~~ ~~1~~ 0
E ~~1~~ 0
F = ~~2~~ ~~1~~ 0
G = ~~2~~ ~~1~~ 0
H = ~~2~~ ~~1~~ 0

queue

~~A~~	~~B~~	~~C~~	~~D~~	~~E~~	F	G H

output A B C D E F G H

Q 38



output

A C D E H F G

queue

~~A~~
~~C~~
~~D~~
~~E~~
~~H~~
~~F~~
~~G~~