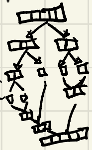
binary search → Time complexity

mergesort
recursion



$O(n \log n)$

worst $n-1$
best $n/2$

$\frac{n^2-n}{2}$ $\frac{1}{2}[(n-1)(n+2)]$ $n-1$

sort unsorted

insertion
loop(swap)

worst n avg $O(n^2)$ best $O(n)$

$\frac{n^2-n}{2}$

Quicksort
recursive (pivot)
L - left Larger
R - right smaller

| S | L | | R | P |

$P > S$ $R < P < L$

$L \leftrightarrow R$

continue till $x > P$
then $x \leftrightarrow P$ + recursion

worst $O(n^2)$ Avg $O(n\log_n)$
best $O(n\log_n)$

heap sort
(ordered binary tree)
Max heap = parent > child
build-max heap = creates max heap from unsorted array
heapify = similar to build-max-heap but part all sorted

$O(n \log_n)$
build-max-heap = $O(n)$
heapify $O(\log n)$, called n-1 times

best = decending order

get max() put at end of list

fix heap $\log_n$
deleteMax $\log_n$
heapifying n
Construct heap n
max heap n
height of heap $\log n$

tut 1 Wk3

Q1) 2 initial arrangement for worst cases for insertion sort:

1) $\{n, n-1, n-2, \ldots, 3, 2, 1\}$

2) $\{1, n, n-1, n-2, \ldots, 2\}$

Q2) $W_n = W_{n/2} + W_{n/2} + 2 + \cancel{W_0} + W_1 + W_2$

obvious

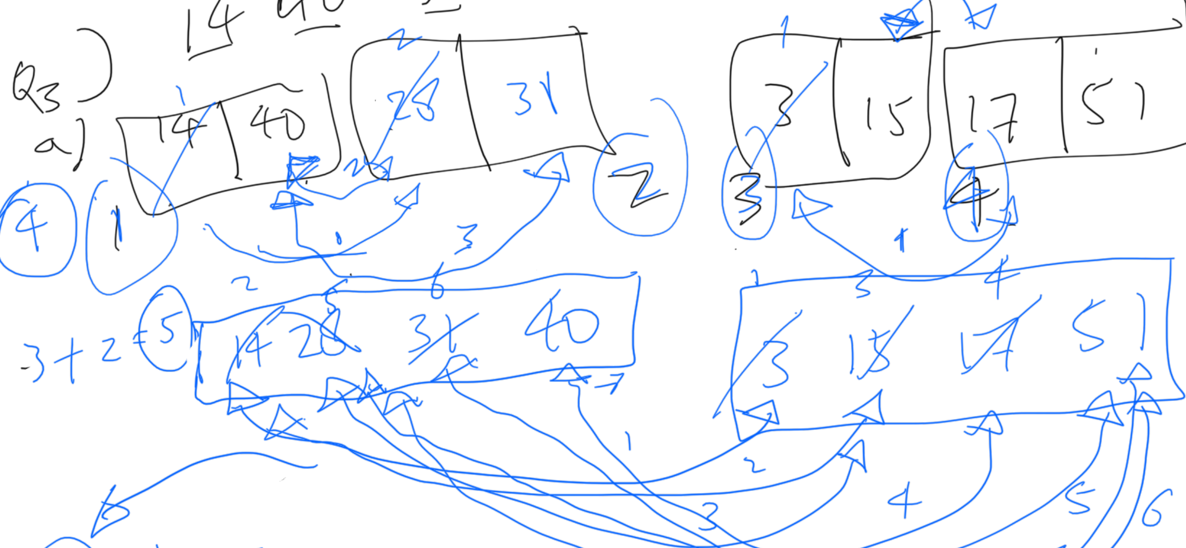
$W_n = 2(W_{n/2}) + 2$

$W_n = 2(2W_{n/2^2} + 2) + 2$

$W_n = 2^2(W_{n/4}) + 2^2 + 2$

14 40 31 28 3 15 17 51

Q3) a)

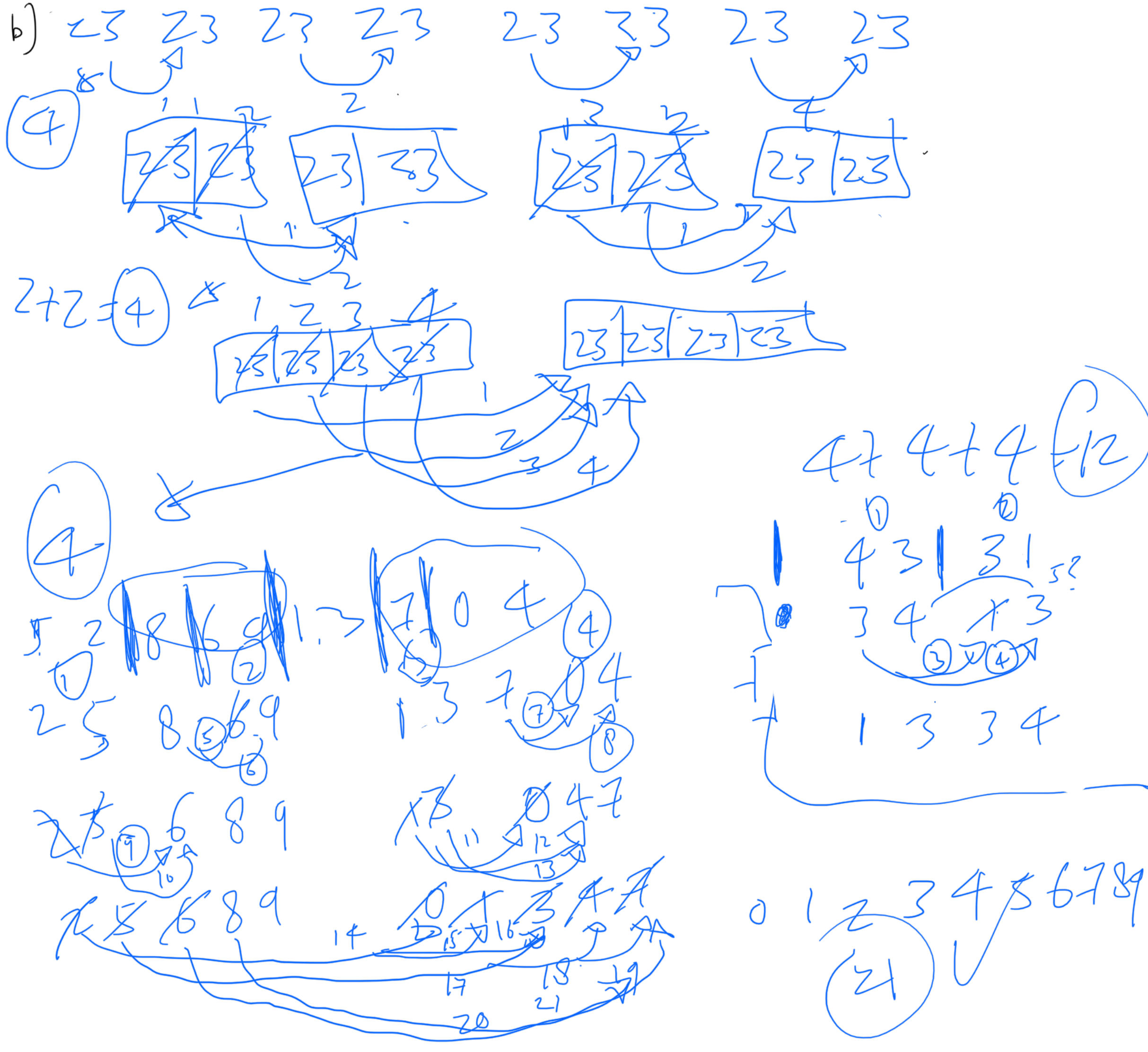


$3 + 2 = 5$

7 3 14 15 17 28 31 40 50

$4 + 5 + 7 = 16$

b)



wk4

Q4) E[middle] median of E[F], $E[(F+L)/2]$, E[L]

TC for E[F] $= n^2$

TC for E[L] $= n^2$

TC for E[mid] $= n \log n$

$$\overbrace{\boxed{\qquad\qquad\qquad\qquad}}^{n}$$

$$(1 + n + n/2)$$

key comparison for choosing pivot = 3

worst case for example if 100, 99, 98 and rest of the element is > 100

Partition will take place in $n-2$ and 2 size of subrange

Q5)

Q6) n elements

a) while element is sorted to a sorted list check with the previous element, if they are the same update a boolean variable to true

b) method is n (1 key comparison)

c)

! worst case

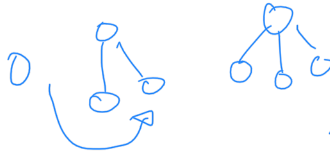
Quick union → Follow Root for Click
Follow parent for id

MN

Quick find   Change whole connected

MN

weighted Quick Union

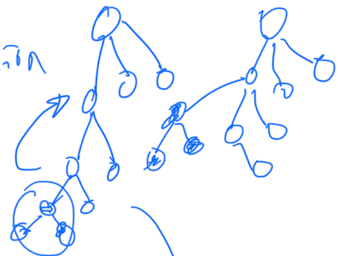


N+M log N

id[i] = parent

weighted Quick union path compression

N+M log* N



init	union	find	connected
O(N)	O(N)	O(N)	O(N)
O(N)	O(N)	O(1)	O(1)
O(N)	O(log N)	O(log N)	O(log N)

Kruskal algo   edge in increasing order

$O(|E| \log |E|)$

TUT WK6

Q1)

	$i=0$	1	2	3	4	5
	S	S	S	S	S	S
1	0	1	1	1	1	1
2	0	0	0	0	1	1
3	0	0	1	1	1	1
4	0	0	0	1	1	1
5	0	0	0	0	0	1

	d	d	d	d	d	d
1	0	0	0	0	0	0
2	∞	4	4	4	4	4
3	∞	2	2	2	2	2
4	∞	6	3	3	3	3
5	∞	8	8	6	5	5

	p_i	p_i				
1	?	?	?	?	?	?
2	?	1	1	1	1	1
3	?	1	1	1	1	1
4	?	1	3	3	3	3
5	?	1	1	4	4	4

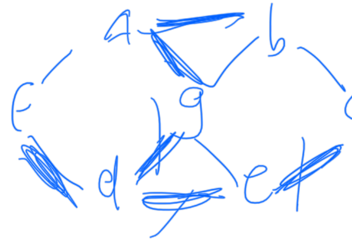
wk7

Q4)

S	0	1	2	3	4	5	6	
a	0	0	0	0	0	1	1	1
b	0	0	0	0	0	1	1	1
c	0	0	0	0	1	1	1	1
d	0	0	1	1	1	1	1	1
e	0	0	0	1	1		1	1
f	0	0	0	0	0	0	0	1
g	1	1	1	1	1	1	1	1

d								
a	∞	3	3	3	3	3	3	3
b	∞	6	6	6	4	2	2	2
c	∞	∞	∞	2	2	2	2	2
d	∞	1	1	1	1	1	1	1
e	∞	3	2	2	2	2	2	2
f	∞	∞	5	5	5	2	5	5
g	0	0	0	0	0	8	0	0

p								
a	n	g	g	g	g	g	g	g
b	n	g	g	g	c	a	a	a
c	n	n	n	e	e	e	e	c
d	n	g	g	g	g	g	g	g
e	n	g	d	d	d	d	d	d
f	n	n	d	d	d	d	d	
g	n	n	n	n	n	n	n	n

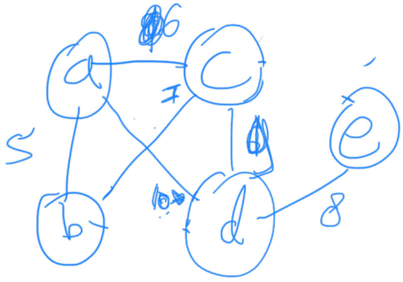


Q5) by contradiction, if the path between undirected vertex a and b in minimum spanning tree is false, there will be path p' that is shorter than a to b that will not form a cycle if added to the graph, however assume there is an alternative path detour a to u, w'[u, b] must be weight less than w[a, b]

however, weight $W(a,b) < W(u,b)$

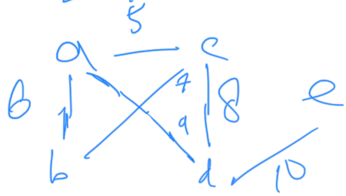
this contradict with the assumtion

Q6)



5, 6, 8, 9 29, yes

7 10



10

5, 6, 7, 8, 9

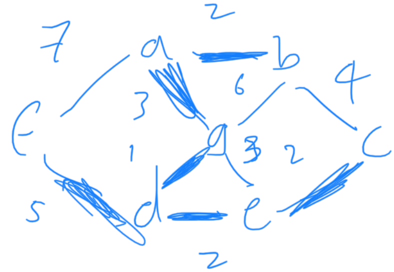
7 10 9

wk8 i = iteration 6 7

Q7)

i	0	1	2	3	4	5	8
id							
a	a	a	a	a	a	d	d
b	b	b	a	a	d	a	d
c	c	c	c	c	d	d	d
d	d	d	d	d	d	d	d
e	e	e	e	d	d	d	d
f	f	f	f	f	f	f	d
g	g	d	d	d	d	d	d

i	0	1	2	3	4	5	8
sz							
a	1	1	2	2	2	2	2
b	1	1	1	1	1	1	1
c	1	1	1	1	1	1	1
d	1	2	2	3	4	6	7
e	1	1	1	1	1	1	1
f	1	1	1	1	1	1	1
g	1	1	1	1	1	1	1



Q9) 1) initialized starting vertex O(1)

2) process every (edge, vertex) by 1 by using UnionFind

3) with method of UnionFind, array that keep track of visited vertex

4) if vertex has been visited return true as it has been visited

$O(|V| + |E| \log^* |V|)$

$W(2) = 1$

$W(n) = 2\,W(n/2) + n - 1$

guess $W(n) = O(n^2)$ $\quad W(n) \le cn^2$

base case $W(2) = 1 \le 2^2$

correct

$W(n/2) = c\left(\frac{n}{2}\right)^2$

$\cancel{2}\, c\left(\frac{n}{\cancel{2}}\right)^2 + \frac{n}{2} - 1 \le cn^2$

hence $W(n) \le cn^2$ is true

if $W(n) \le cn$

$1 \le 2.c$ correct

$W(n/2) = \frac{n}{2}c$