19BCE245 2CS301 3 22/12/2020 1

Q-1 A [A - B (B + ((C ^ D ^ E) * F - G) / ((H - I) * J - K)]

steps	Expression	stack	Postfix
0	A	(	A
1	-	-	A
2	(	(-	A
3	B	(-(	AB
4	+	(-(	AB
5	(	(-(+	AB
6	C	(-(+(	ABC
7	^	(-(+(^	ABC
8	D	(-(+(^	ABCD
9	^	(-(+(^^	ABCD
10	E	(-(+(^^	ABCDE
11	)*	(-(+*	ABCDE^^
12	F	(-(+*	ABCDE^^F
13	-	(-(-	ABCDE^^F*+
14	G	(-(-	ABCDE^^F*+G
15	)	(-	ABCDE^^F*+G-
16	)	(-/	ABCDE^^F*+G-
17	/	(-/(	ABCDE^^F*+G-
18	(	(-/((	ABCDE^^F*+G-
19	H	(-/((	ABCDE^^F*+G-H
20	-	(-/((-	ABCDE^^F*+G-H
21	I	(-/((-	ABCDE^^F*+G-HI
22	)	(-/(	ABCDE^^F*+G-HI-
23	*	(-/(*	ABCDE^^F*+G-HI-
24	J	(-/(*	ABCDE^^F*+G-HI-J
25	-	(-/(-	ABCDE^^F*+G-HI-J*
26	K	(-/(-	ABCDE^^F*+G-HI-J*K
27	)	(-/	ABCDE^^F*+G-HI-J*K-

vision

19BCE245 2CS301 3 22/12/2020 2 Layout

28.) ABCDE^^F *+G-HI-J*K-/-

29)

• final postfix: ABCDE^^F*+G-HI-J*K-/-

Q-1

(B.) [2:8, -4:5, 6:15]

memory address of [5, 3, 11] = (?) [column major]

base add. = 1000

w = 4.

here

L_1 = Upper bound - lower bound + 1 = 8 - 2 + 1 = 7

L_2 = Upper bound - lower bound + 1 = 5 + 4 + 1 = 10

L_3 = Upper bound - lower bound + 1 = 15 - 6 + 1 = 10.

also.

E_1 = K_1 - lower bound = 5 - 2 = 3

E_2 = K_2 - lower bound = 3 - (-4) = 7

E_3 = K_3 - lower bound = 11 - 6 = 5

∴ for column major

Address = $B + W((E_3 * L_2 + E_2)L_1 + E_1)$

= $1000 + 4(((5*10) + 7)*7 + 3)$

= $1000 + 4(57*7 + 3)$

= $1000 + 4(399 + 3)$

= $1000 + 1608$

= 2608

19BCE2445 2CS301 3 7/11/2020 (3)

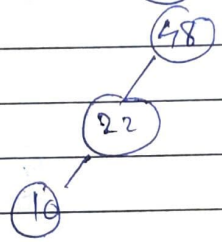
Q-2 A 48, 22, 10, 5, 8, 4, 33, 21, 9, 13

creating AVL tree:

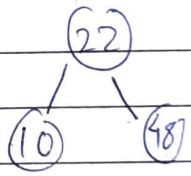
→ inserting 48 →
48 as root node

→ inserting 22 →
22 < 48

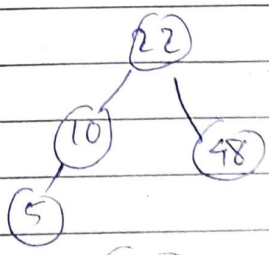
→ inserting 10 →
10 < 48
10 < 22



here, LL rotation:
after

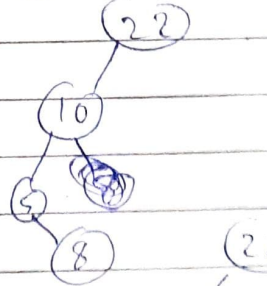


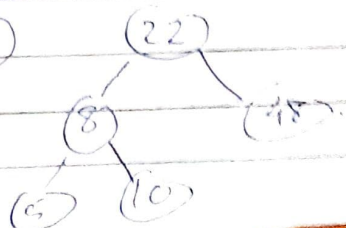
→ inserting 5 →
5 < 22
5 < 10



→ inserting 8 →
8 < 22
8 < 10.

here after LR rotation:





VISION

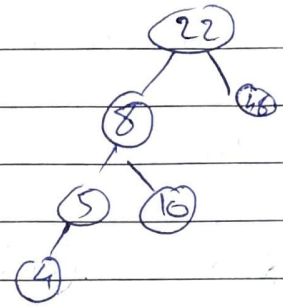
19BCE2145 2CS301 3 22/12/2020 (4) Sanyal

→ inserting 4.

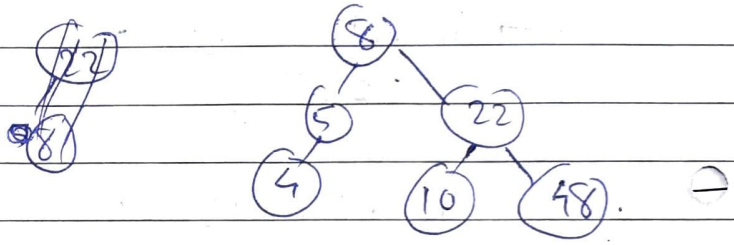
4 < 22

4 < 8

4 < 5



here after LL rotation:

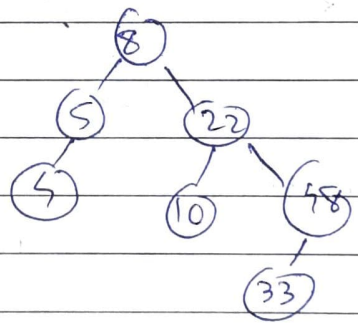


→ inserting 33

33 > 8

33 > 22

33 < 48

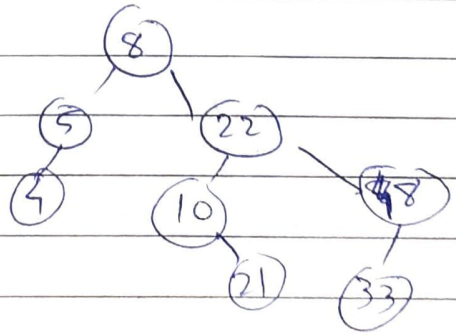


~~after~~ → inserting 21

21 > 8.

21 < 22.

21 > 10

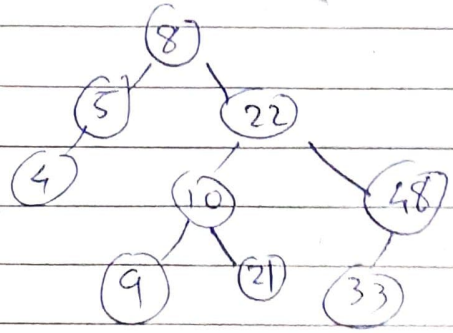


→ inserting 9.

9 > 8

9 < 22

9 < 10



vision

19BCE245 2CS301 3 22/12/2020

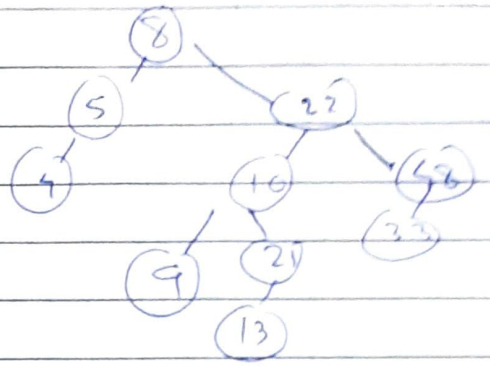
inserting 13 →

13 > 8

13 < 22

13 > 10

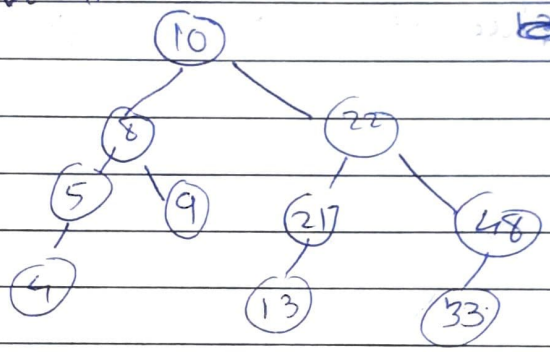
13 < 21



performing RL rotation

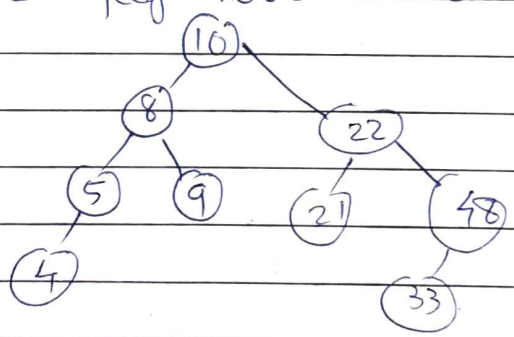
final AVL tree:

→



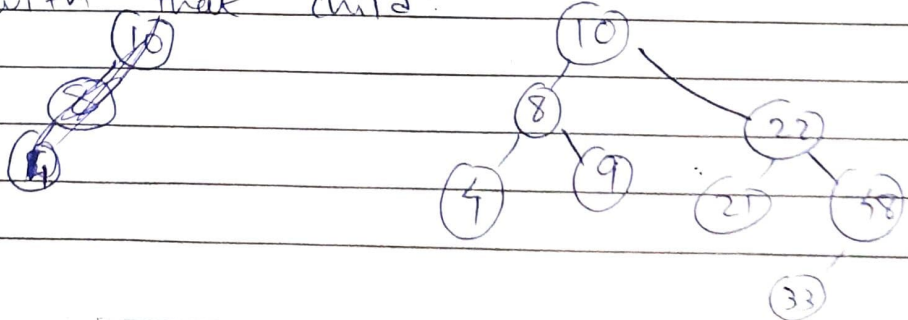
→ deleting 13

its a leaf node. so directly removing it



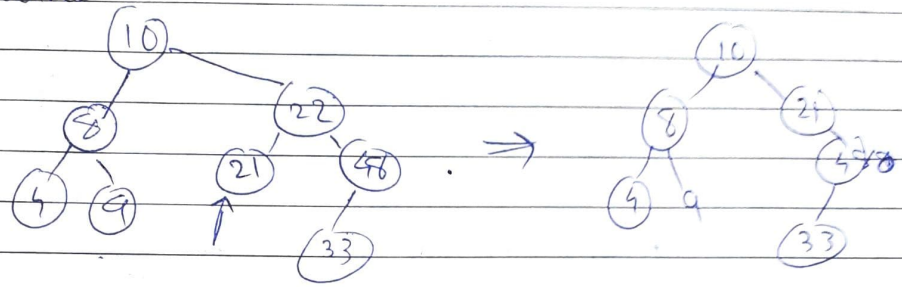
→ Deleting 5

its has one child so replacing deleted node with that child.



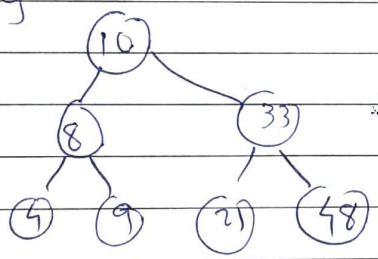
→ deleting 22 →

It has two child so finding max from its left subtree.



max = 22 → replacing and performing RL rotation →

resulting tree.



Q-2 B.

In order = alice, bill dave, fred, jane, joe, judy, marry, tom.

Post order = alice, dave, joe, jane, fred, bill, tom, marry, judy

considering alice as root nod

19BCE245 2CS301 3 22/12/2020 (12)

19BCE245 2CS301 3 22/12/2020 (7) (8)

19BCE245 2CS301 3 22/12/2020 (8) days

Q-3 A) Keys = 5, 25, 15, 35, 95.

$h'(x) = (h1(x) + i * h2(x)) \% 10$

$h1(x) = x \% 13$ $h2(x) = 7 - (x \% 7)$

→ 5

$h1(5) = 5 \% 13 = 5$

$h2(5) = 7 - (5 \% 7) = 2$

$h'(5) = (h1(5) + i \cdot h2(5)) \% 10.$

$= (5 + 0 \cdot 2) \% 10$ $(\because i = 0)$

$= 5$

→ 25

$h1(25) = 25 \% 13 = 12$ $h2(25) = 7 - (25 \% 7) = 3$

$h'(25) = (h1(25) + i \cdot h2(25)) \% 10.$

$= (12 + 1 \cdot 3) \% 12.$

$= 15 \% 12 = 3$

→ 15

$h1(15) = 15 \% 13 = 2$ $h2(15) = 7 - (15 \% 7) = 6$

$h'(15) = (2 + 2 \cdot 6) \% 10$

$= 4$

→ 35

$h1(35) = 35 \% 13 = 9$ $h2(35) = 7 - (35 \% 7) = 7$

$= (9 + 3 \cdot 7) \% 10$

$= 30 \% 10$

$= 0.$

→ 95

$h1(95) = 95 \% 13 = 4$ $h2(95) = 7 - (95 \% 7)$

$= 3$

$h'(95) = (4 + 4 \cdot 3) \% 10$

$= 16 \% 10 = 6.$

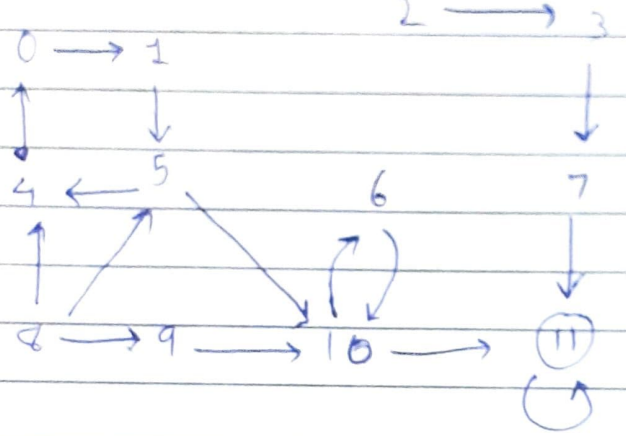
vision

19BCE245

hash table : (using linear probing)

0	1	2	3	4	5	6	7	8	9
35			25	15	5	95			

3 B) DFS traversal :



→ starting from 0, as in DFS we will use stack.

stack : 0

~~inserting~~ 0's all neighbour : 1,

traversal : 0 stack : 1

→ 1's ~~no~~ adjecent nodes : 5 (inserting it in stack)

traversal : 0,1 stack = 5

(after popping 1 from stack.

→ 5' adj nodes : 4, 10

traversal : 0,1,5 stack : ~~4,~~ 10, 4

→ 4's adj nodes : 0 (which is already visited.

traversal : 0, 1, 5, 4 stack : 10

→ 10's adj nodes : 6, 11

traversal : ~~10~~ 0,1,5,4,10. stack : 6, 11

→ 11's adj nodes : 11 (ignoring).

traversal : 0,1,5,4,10,11 stack : 6

vision

19BCL245 2CS301 3 22/12/2020

→ 6's adj nodes: 10 (which is already traversed)
traversal: 0, 1, 5, 4, 10, 11, 6 stack empty.

Q-3B

	A	B	C	D	E	F	G	H
A			6 ✓	5 ✓	7 ✓			
B				3 ✓		8 ✓		
C	6 ✓					7 ✓		
D	5 ✓	3 ✓						2 ✓
E	7 ✓						4 ✓	
F		8 ✓	7 ✓				6 ✓	7
G					4 ✓	8 ✓		
H				2 ✓		7		

edges in increasing order: 2 : HD
3 : DB
4 : GE
5 : AD
6 : AC
7 : FC, AE
8 : FB, FG

Edges collected.

① HD

② HD, DB

③ HD, DB, GE

Initially sets of vertices: {(A), (B), (C), (D), (E), (F), (G), (H)}
→ for Edge HD: {(A), (B), (C), (D, H), (E), (F), (G)}
as they (H and D) are in different sets.
→ for edge 3 : DB.
{(A), (B, D, H), (C), (E), (F), (G)}
as they are in diff sets.
→ for edge GE
{(A), (B, D, H), (C), (G, E), (F)}

vision

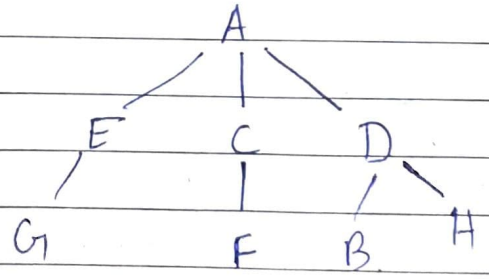
(4) HD, DB, GE, AD → for Edge: AD,
{ (A, B, D, H), (C), (G, E), (F) }
as A and D are in diff sets

(5) HD, DB, GE, AD, AC → for Edge AC
{ (A, B, C, D, H), (G, E), (F) }
as A and C are in diff sets

(6) HD, DB, GE, AD, AC, FC → for Edge FC
{ (A, B, C, D, F, H), (G, E) }
as F and C are in diff sets

(7) HD, DB, GE, AD, AC, FC, AE → for Edge AE
{ (A, B, C, D, F, G, E, H) }
as E and A are in diff sets

→ minimum spanning tree:



4(A) priority queue

→ insertion algo:
If there is no node,
create a newNode
else (a node is already present)
insert the newNode at the end
(last node from left to right)

19BCE245 2CS301 3 22/12/2020 changed

19BCE245 2CS301 3 22/12/2020 changed

12

heapify the array

⟹ for linked list : Insertion algo:

PUSH(HEAD, DATA, PRIORITY)

Step 1 : ~~Cla~~ create new node with DATA and PRIORITY

step 2 : Check if 'head' has lower priority.
If true follow Step 3 & 4 and end. Else go to Step 5

~~st~~ 3 : NEW → NEXT = HEAD

4 : HEAD = NEW

5 : set temp to head of the list.

6 : while temp → next != NULL and temp → Next → priority > priority;

7 : temp = temp → next
[END of LOOP]

8 : New → Next = Temp → Next

9 : Temp → Next = New

10 : END

Algo for POP (deletion:

→ Step 2 : set the head of the list to the next node in the list.
head = head → Next

Step 3 : Free the node at the head of the list

4 : End

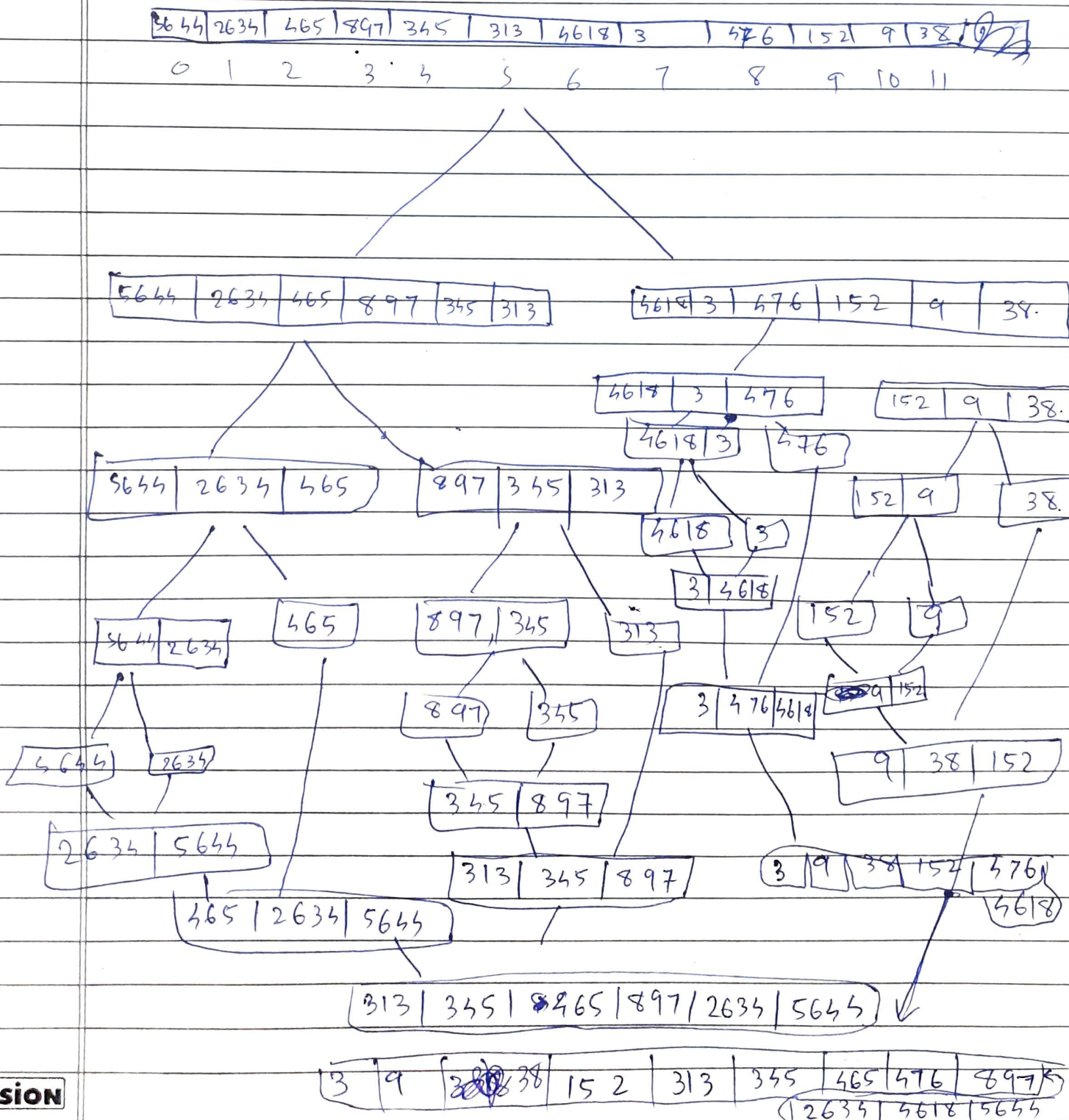
19BCE245 2CS301 3 22/12/2020 daywd

13

Q-4

(B) 5644, 2634, 465, 897, 345, 313, 4618, 3, 476, 152, 9, 38.

: Sorting algo used: merge sort



vision

19BCE2445 2CS301 3 22/12/2020 (14)

Sorted array:

3, 9, 38, 152, 313, 345, 465, 476, 897
2634, 4618, 5644.