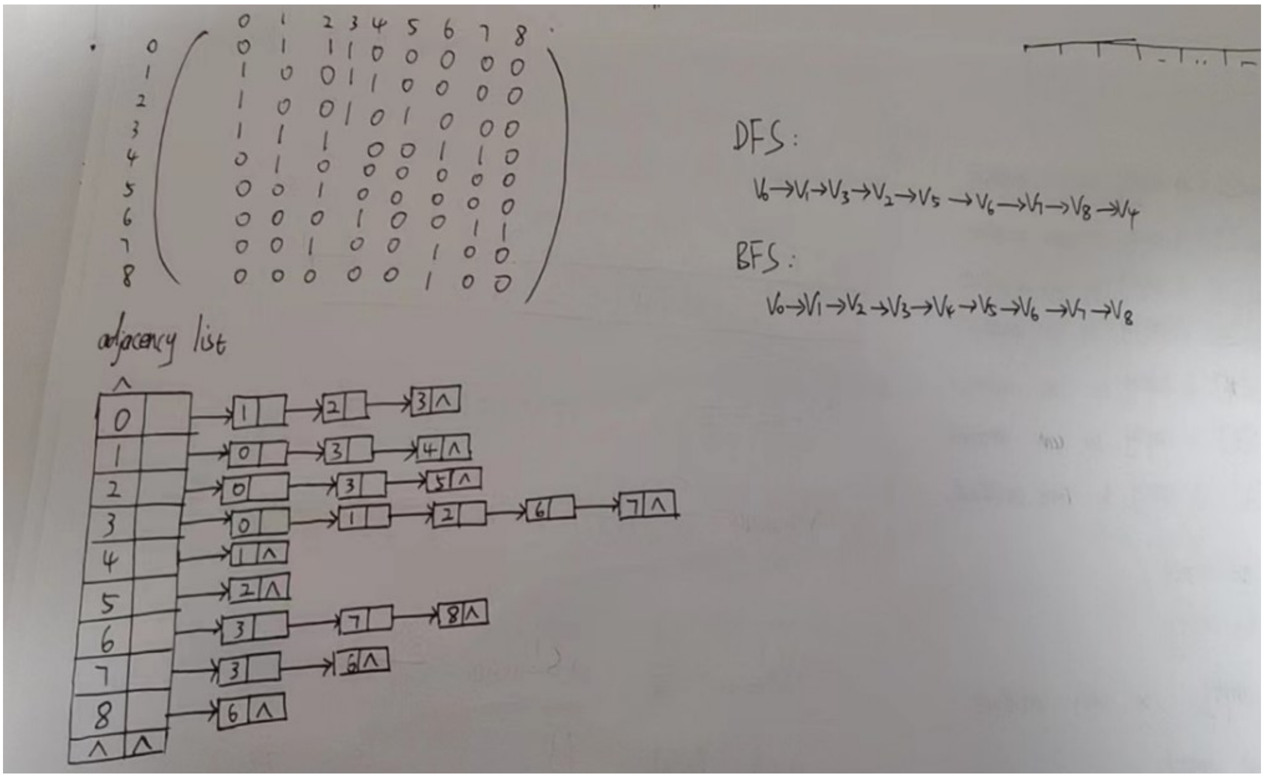


$$\begin{array}{c|ccccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 3 & 1 & 1 & 1 & & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 5 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \\ 6 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & \\ 7 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & \\ 8 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \end{array}$$

adjacency list

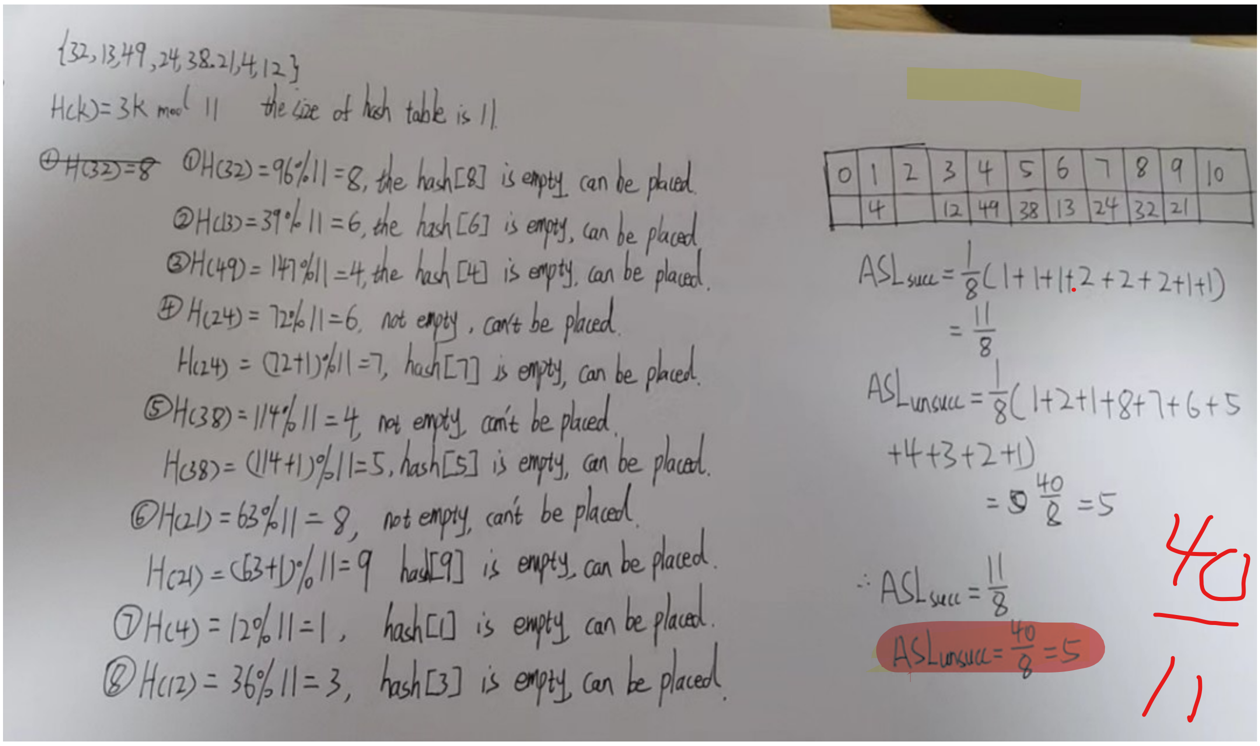
0 → 1 → 2 → 3 ∧
1 → 0 → 3 → 4 ∧
2 → 0 → 3 → 5 ∧
3 → 0 → 1 → 2 → 6 → 7 ∧
4 → 1 ∧
5 → 2 ∧
6 → 3 → 7 → 8 ∧
7 → 3 → 6 ∧
8 → 6 ∧
∧ ∧

DFS:

$V_0 \to V_1 \to V_3 \to V_2 \to V_5 \to V_6 \to V_7 \to V_8 \to V_4$

BFS:

$V_0 \to V_1 \to V_2 \to V_3 \to V_4 \to V_5 \to V_6 \to V_7 \to V_8$



{32, 13, 49, 24, 38. 21, 4, 12}

H(k) = 3k mod 11 the size of hash table is 11.

~~① H(32) = 8~~ ① H(32) = 96%11 = 8, the hash[8] is empty can be placed.

② H(13) = 39%11 = 6, the hash[6] is empty, can be placed

③ H(49) = 147%11 = 4, the hash[4] is empty, can be placed.

④ H(24) = 72%11 = 6, not empty, can't be placed.

H(24) = (72+1)%11 = 7, hash[7] is empty, can be placed.

⑤ H(38) = 114%11 = 4, not empty can't be placed.

H(38) = (114+1)%11 = 5, hash[5] is empty, can be placed.

⑥ H(21) = 63%11 = 8, not empty, can't be placed.

H(21) = (63+1)%11 = 9 hash[9] is empty, can be placed.

⑦ H(4) = 12%11 = 1, hash[1] is empty can be placed.

⑧ H(12) = 36%11 = 3, hash[3] is empty can be placed.

0	1	2	3	4	5	6	7	8	9	10
	4		12	49	38	13	24	32	21	

$ASL_{succ} = \frac{1}{8}(1+1+1+2+2+2+1+1)$

$= \frac{11}{8}$

$ASL_{unsucc} = \frac{1}{8}(1+2+1+8+7+6+5$

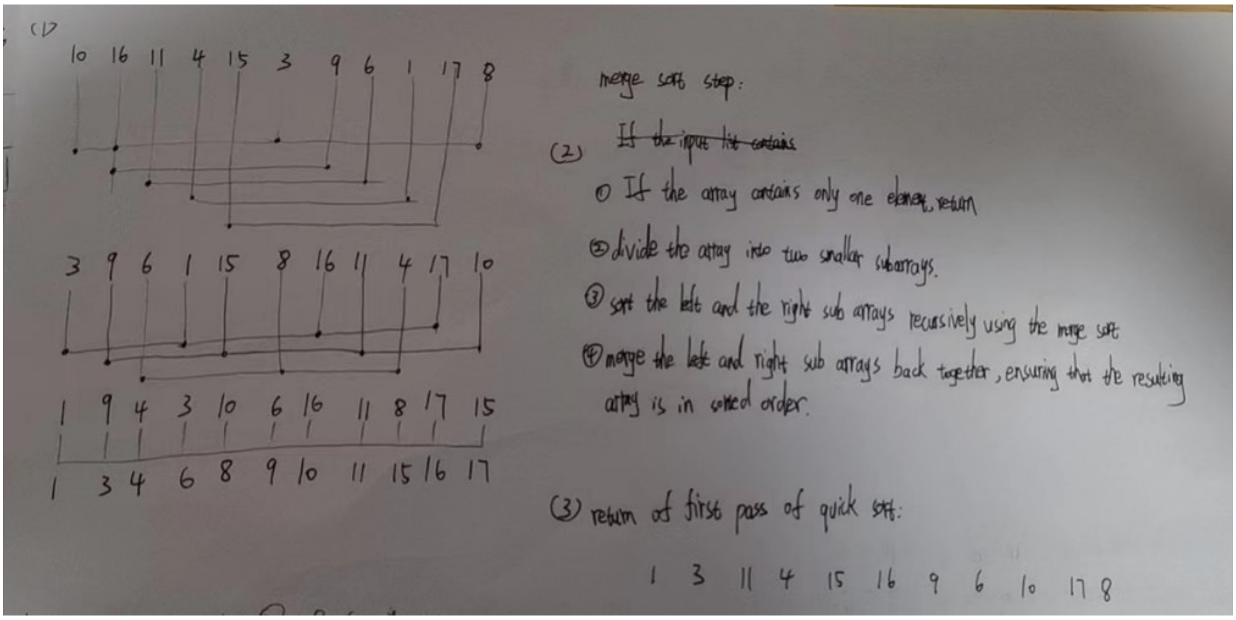
$+4+3+2+1)$

$= \frac{40}{8} = 5$

$\therefore ASL_{succ} = \frac{11}{8}$

$ASL_{unsucc} = \frac{40}{8} = 5$

$\frac{40}{11}$ −



(1)

10 16 11 4 15 3 9 6 1 17 8

3 9 6 1 15 8 16 11 4 17 10

1 9 4 3 10 6 16 11 8 17 15

1 3 4 6 8 9 10 11 15 16 17

merge sort step:

(2) ~~If the input list contains~~

① If the array contains only one element, return

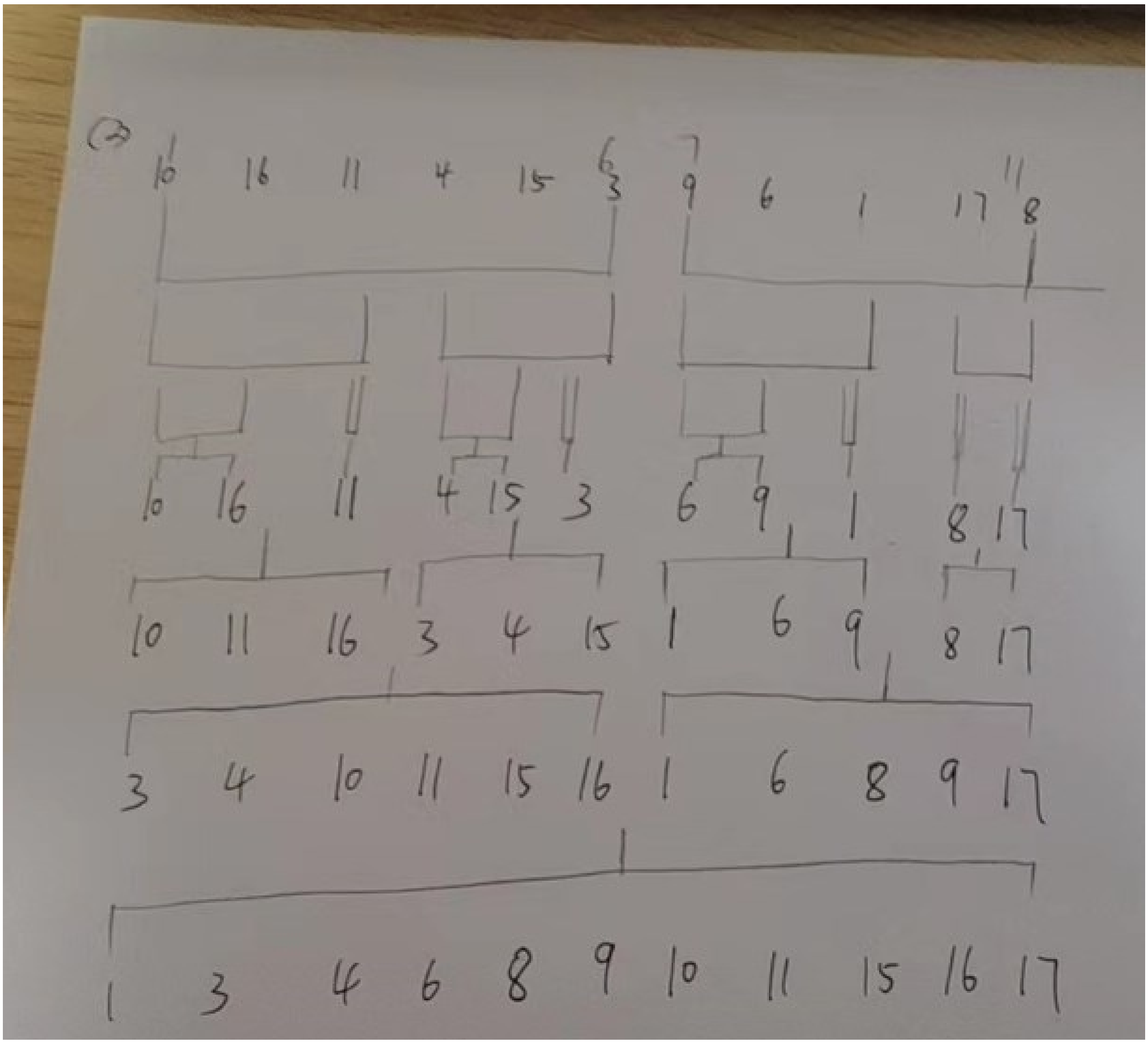
② divide the array into two smaller subarrays.

③ sort the left and the right sub arrays recursively using the merge sort

④ merge the left and right sub arrays back together, ensuring that the resulting array is in sorted order.

(3) return of first pass of quick sort:

1 3 11 4 15 16 9 6 10 17 8



(2)

1 6 7 11

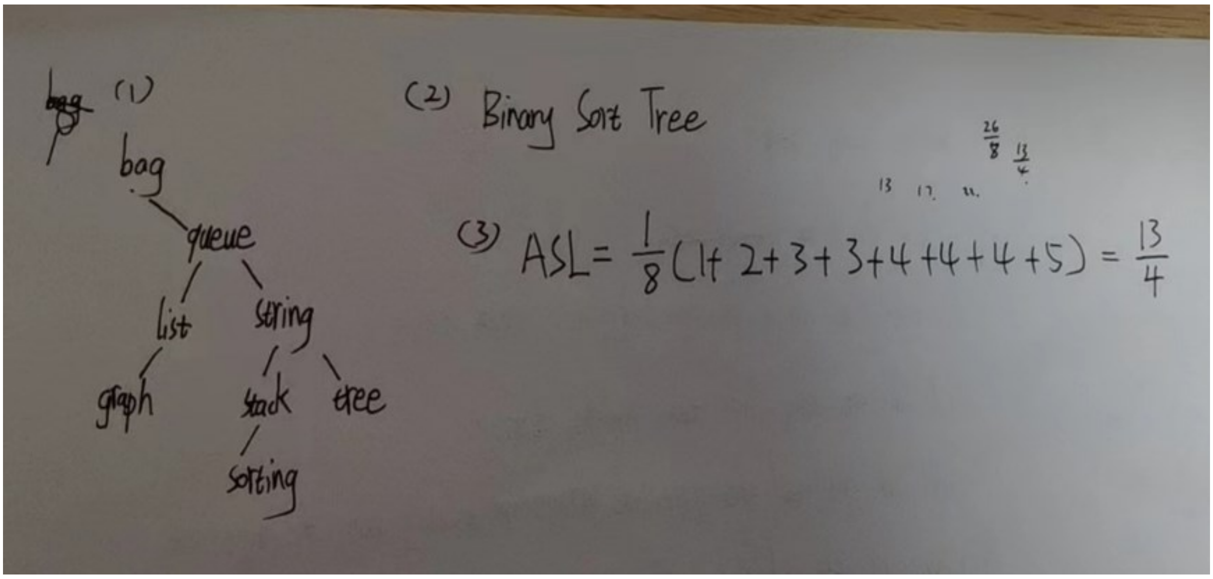
10 16 11 4 15 3 9 6 1 17 8

10 16 11 4 15 3 6 9 1 8 17

10 11 16 3 4 15 1 6 9 8 17

3 4 10 11 15 16 1 6 8 9 17

1 3 4 6 8 9 10 11 15 16 17



~~bag~~ (1)

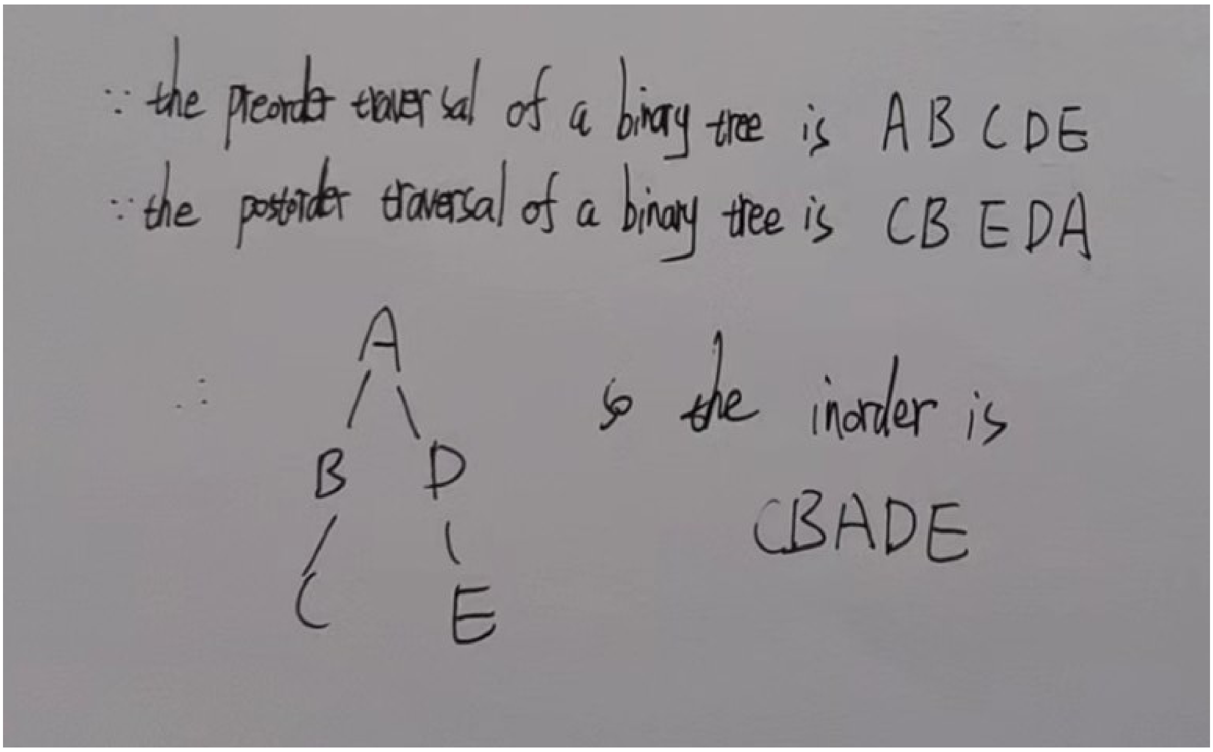
bag
queue
list
string
graph
stack
tree
sorting

(2) Binary Sort Tree

$\frac{26}{8}$ $\frac{13}{4}$

13 17. 11.

(3) $ASL = \frac{1}{8}(1+2+3+3+4+4+4+5) = \frac{13}{4}$



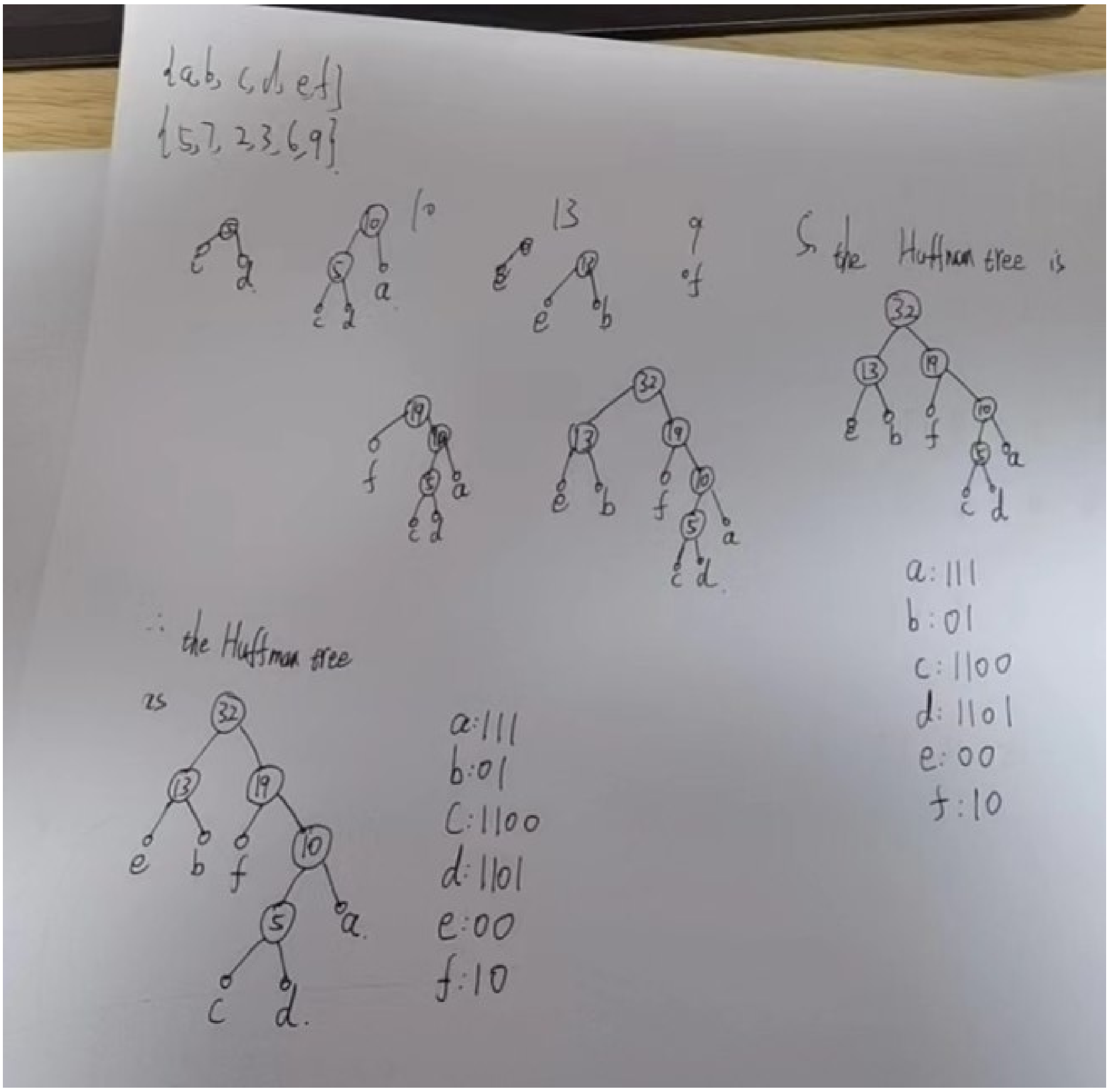
∵ the preorder traversal of a binary tree is A B C D E

∵ the postorder traversal of a binary tree is C B E D A

∴

so the inorder is

CBADE



{a,b, c,d, e,f}
{5,7, 2,3, 6,9}

∴ the Huffman tree is

a: 111
b: 01
c: 1100
d: 1101
e: 00
f: 10

∴ the Huffman tree is

a:111
b:01
C:1100
d:1101
e:00
f:10