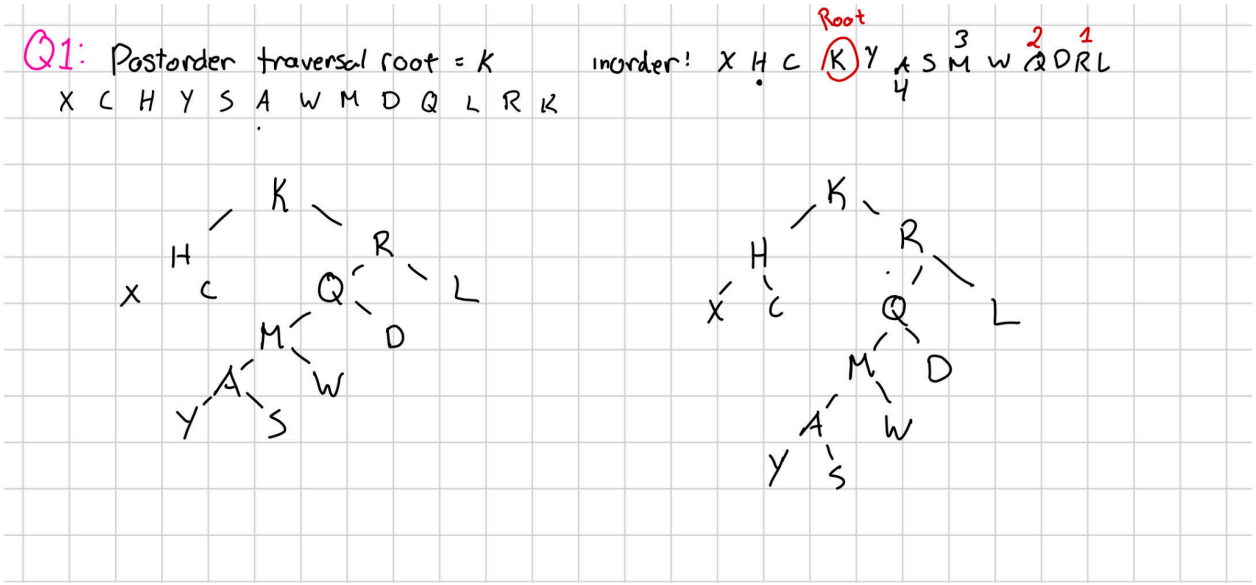
Steven Ly - 40215469

Theory #3

Question 1:



Yes, it is possible to construct a binary tree with the given traversals. The **root** is the last element of the postorder traversal **K**, and its position in the inorder traversal splits the tree into left **X H C** and right **Y A S M W Q D R L** subtrees.

The root of each subtree is the last element of its segment in the postorder traversal. Recursively, the left subtree of **B** is rooted at **H** with children **X** and **C**, and the right subtree is rooted at **R**, with **Q** and **L** as its children. Continuing this process identifies the full tree, satisfying both traversals.

Question 2:

(i)
Advantages:

- Space-efficient: No need for a special **`AVAILABLE`** marker.
- Search for removed keys terminates faster (as soon as their negative value is found).

Disadvantages:

- Prevents reuse of the slot for future insertions. Negative markers block valid keys from being inserted into the table.
- Complicates the search process since the algorithm must check for negative values.

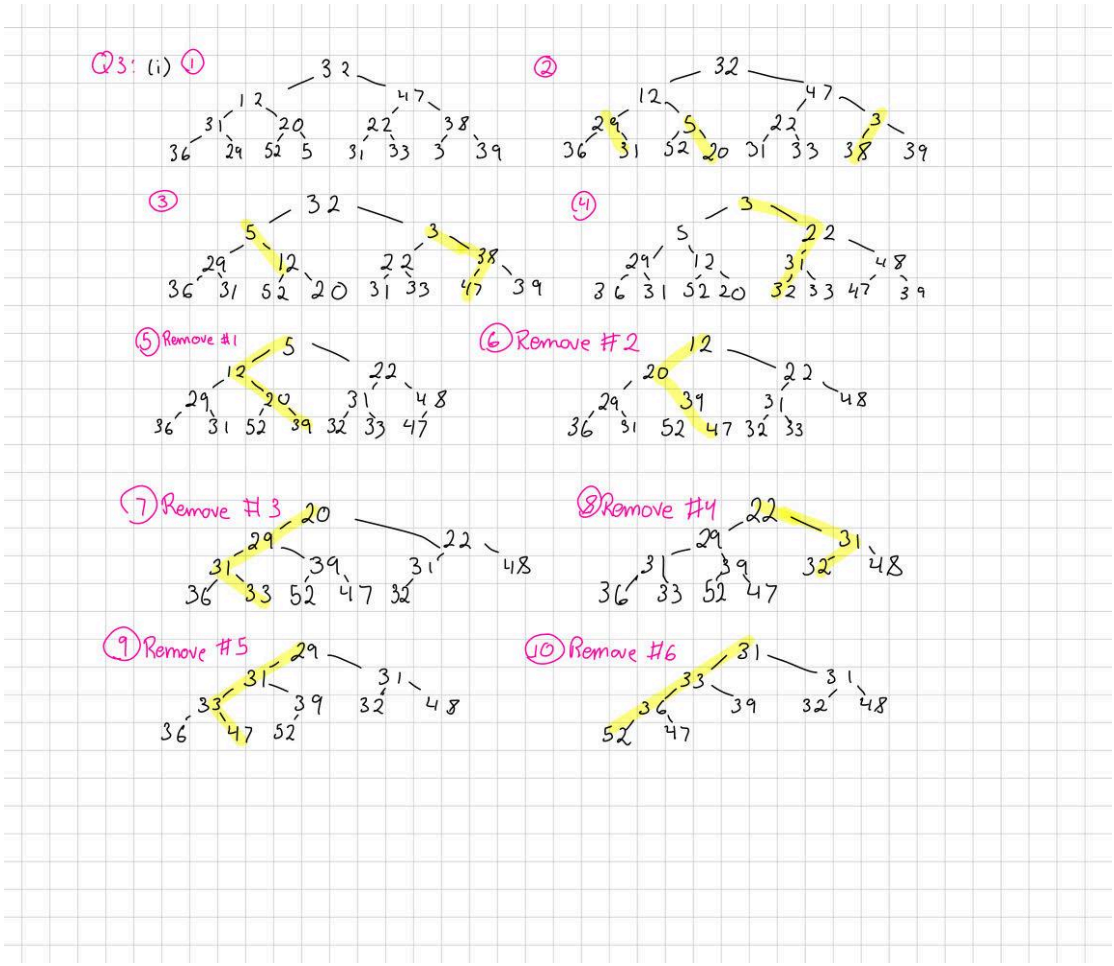
Misbehavior:

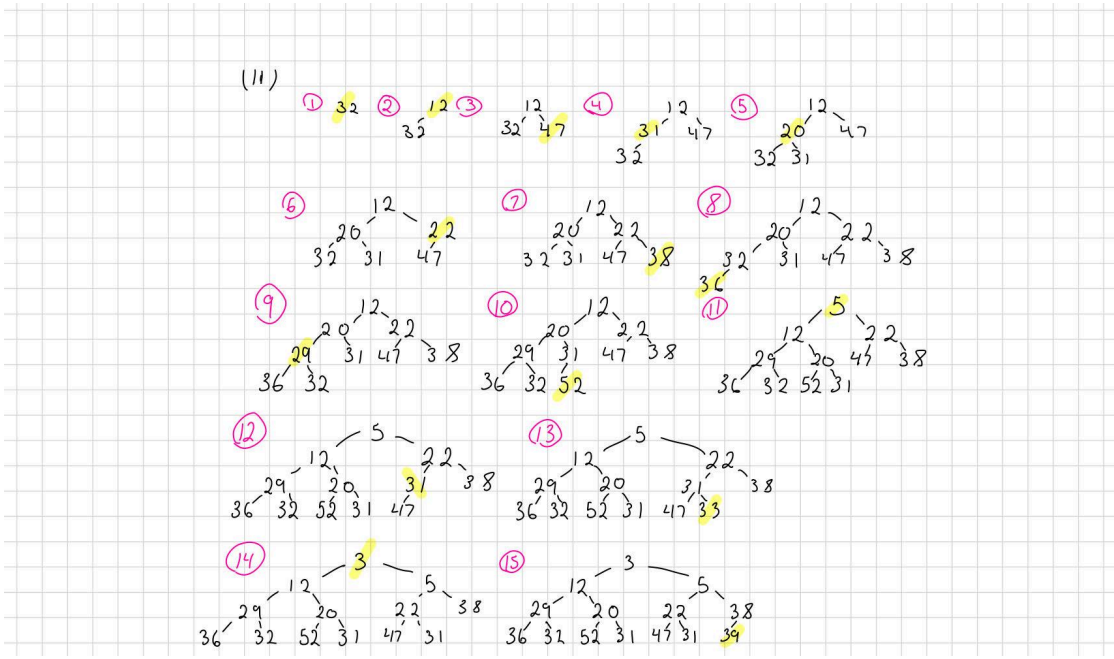
- If a negative marker remains in the table, it prevents proper handling of future collisions, potentially causing the table to fill up incorrectly.

(ii)

- **Advantages**:
    - Improves search efficiency by keeping keys closer to their original hashed locations.
    - Avoids the need for a special **`AVAILABLE`** marker.
- **Disadvantages**:
    - Increases the complexity of deletion and insertion.
    - Disrupts the probing sequence for other keys, making some keys unreachable during searches or creating errors during future insertions.
- **Misbehavior**:
    - Relocating keys can break the linear probing sequence. This leads to cases where valid keys become unreachable because the probing path is altered.

Question 3:





Question 4:

Idx	0	1	2	3	4	5	6	7	8	9	10	11	12
	182	352	93	120	56	18		33	21	140			51
	91				472	70				217			90
	78									204			
										178			

Slot 9 has the maximum amount of collisions of 3

Question 5:

a)

0	23	32	72	76	22	73	40	30	20	60	16	74	28	14
1	23	32	72	22	73	40	30	20	60	16	74	28	14	76
2	23	32	22	72	40	30	20	60	16	73	28	14	74	76
3	23	32	22	40	30	20	60	16	72	28	14	73	74	76
4	23	22	32	30	20	40	16	60	28	14	72	73	74	76
5	22	23	30	20	32	16	40	28	14	60	72	73	74	76
6	22	23	20	30	16	32	28	14	40	60	72	73	74	76
7	22	20	23	16	30	28	14	32	40	60	72	73	74	76
8	20	22	16	23	28	14	30	32	40	60	72	73	74	76
9	20	16	22	23	14	28	30	32	40	60	72	73	74	76
10	16	20	22	14	23	28	30	32	40	60	72	73	74	76
11	16	20	14	22	23	28	30	32	40	60	72	73	74	76
12	16	14	20	22	23	28	30	32	40	60	72	73	74	76
13	14	16	20	22	23	28	30	32	40	60	72	73	74	76

b) For Bubble sort The Worst case is $O(n^2)$ if the array is completely reverse sorted and the best case is O(n) if the array is already sorted. For Selection Sort, the time complexity is always $O(n^2)$. Bubble sort can perform better than selection sort when the array is nearly sorted due to

its ability to terminate early if no swaps are needed. However, on average both have similar time complexity of $O(n^2)$.

Question 6:

0	491	692	992	472	182	543	783	264	295	356	947
1	543	947	356	264	472	182	783	491	692	992	295
2	182	264	295	356	472	491	543	692	783	947	992

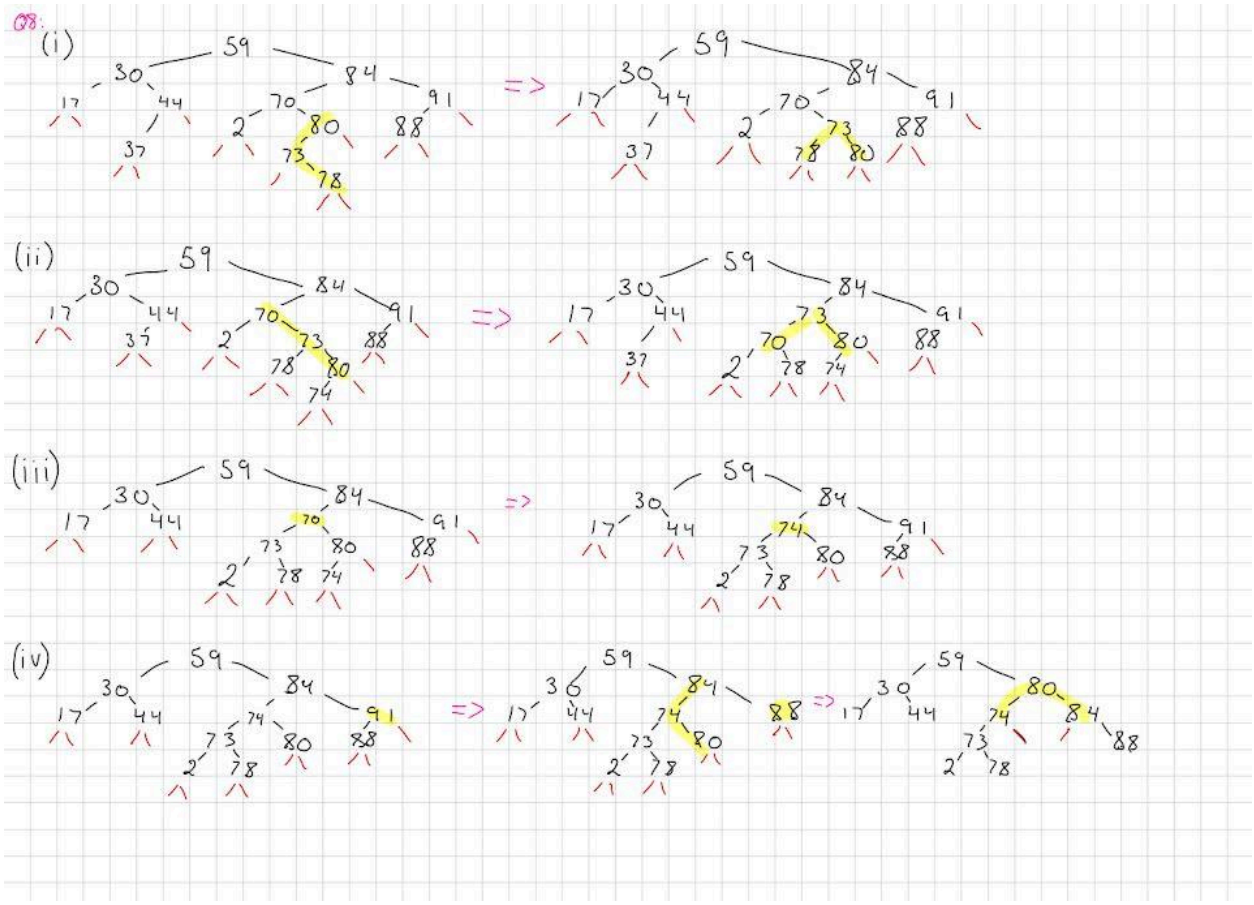
Question 7:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
38	39			DEL		25	45			29		DEL			88	35		18

(i) The largest cluster is 1, as there are no consecutive filled slots in the table
(ii) 88 collided at slot 12 and was place in slot 15, Duplicate put(29) caused no additional collision as the key was already present
(iii) $Load\ Factor\ =\ Number\ of\ Occupied\ Slots\ \div\ Table\ Size$

$Load\ Factor\ =\ 8\ \div\ 19\ \approx\ 0.42$

Question 8:



(ii)Time Complexity is O(logn)
(iii)Time Complexity is O(logn)
(iv)Time Complexity is O(logn)

Question 9:

Node	Distance	Previous Node
N	0	-
A	8	N
W	3	N
D	1	N
E	8	M

T	7	M
M	5	D
K	14	M