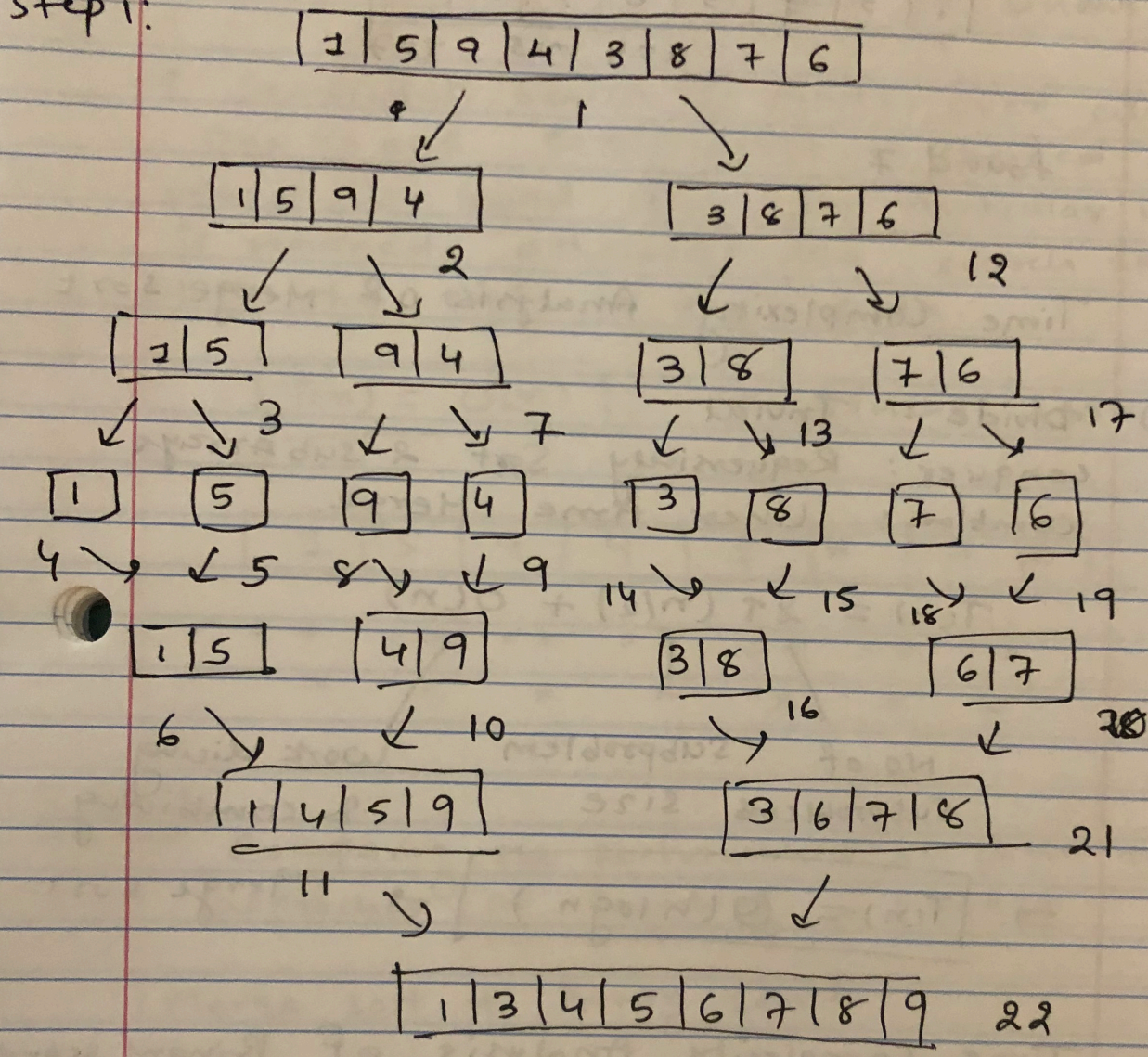
Q.9

Merge sort

step 1:



Binary Search and find number : 7

0	1	2	3	4	5	6	7
1	3	4	5	6	7	8	9

l=0 m=3 r=7

step 1: $m = l + (r-l)/2 = 0 + 3.5 = 3$

0	1	2	3	4	5	6	7
1	3	4	5	6	7	8	9

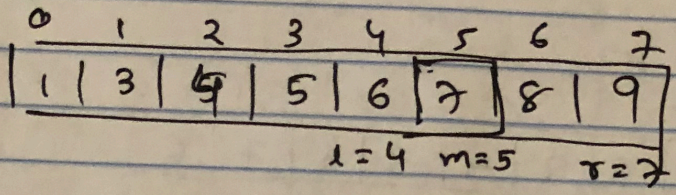
step 2: l=4 r=7

775

$m = l + (r-l)/2 = 4+(7-4)/2 = 5$

$m = l + (r-l)/2 = 4+1.5 = 2.5$ ~~struck~~

Step 3:



$7 = m$ found 7

Time Complexity Analysis of Merge sort

1) Divide : Trivial
2) Conquer : Requersively Sort 2 subArrays
3) Combine : Liner time Merge.

$$T(n) = 2T(n/2) + O(n)$$

- No of subproblems
- Subproblem size
- Work diving & combining.

$\Rightarrow$ $\boxed{T(n) = \Theta(n \log n)}$ → Merge sort

Time Complexity Analysis of Binary search.

1) Divide: Trivial
2) Conquer : Search given number in Sorted subproblem.

$$T(n) = 1T(n/2) + O(1)$$

- no of subproblems
- Subproblem size
- Work dividing & combining.

$T(n) = O(\log n)$ → Binary search.

Step 2

Time complexity Analysis of Linear Search

→ Sequential search is made over all items one by one. every items is checked and if a match is found then the particular items is returned otherwise the search continues till the end of the data.

$T(n) = O(n)$ → Linear search

1	5	9	4	3	8	7	6

$=7$ $=7$ $=7$ $=7$ $=7$ $=7$ $=7$

× × × × × × ✓

Step 3

Comparing the performance of these two approches in searching a number.

Merge sort + Binary search

$T(n) = O(n \log n) + O(\log n)$

$= O(n \log n)$

Linear Search

$T(n) = O(n)$

Conclusion: Even though Merge Sort + Binary search takes $O(n \log n)$ but subsequent search is will take $O(\log n)$ which is faster than liner search $O(n)$.