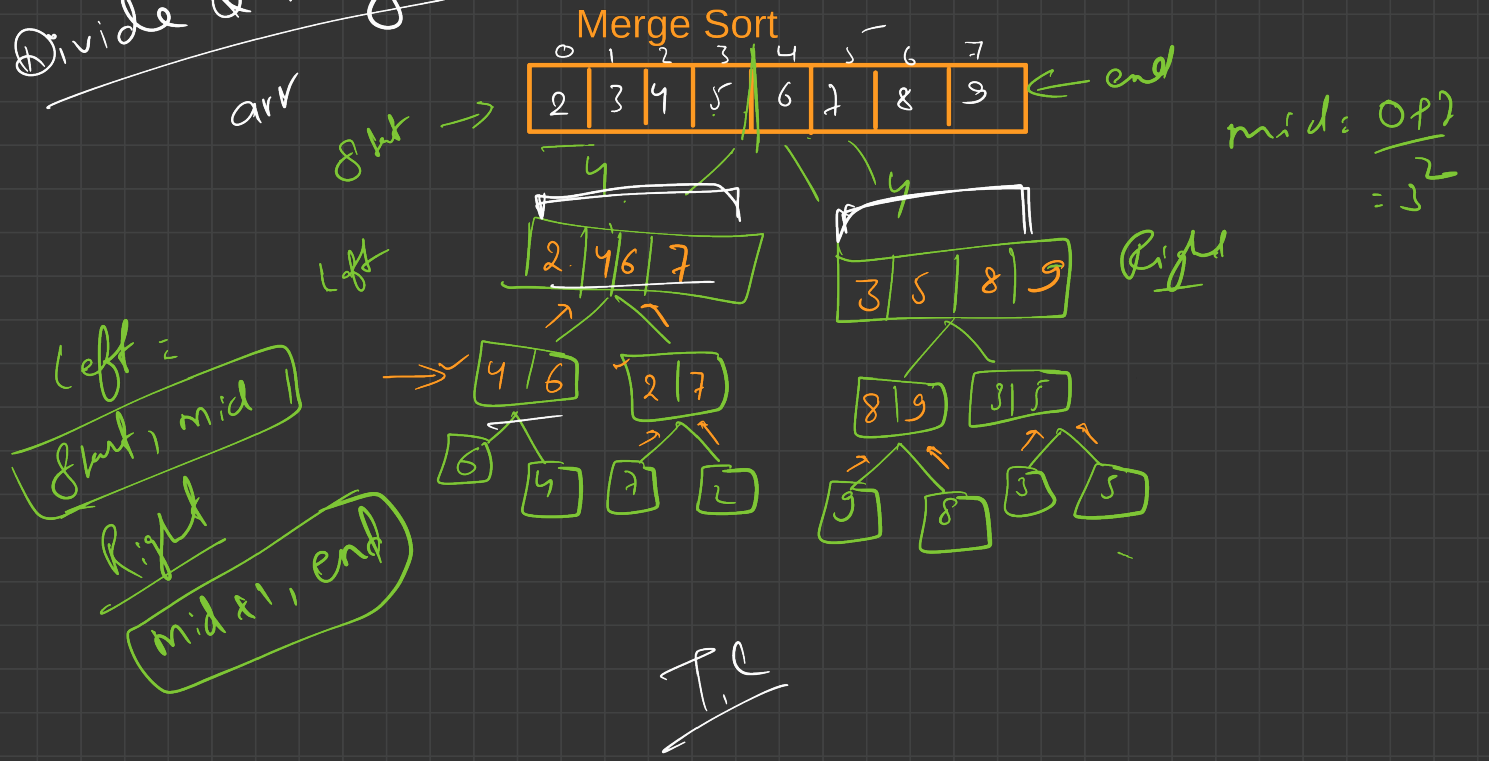
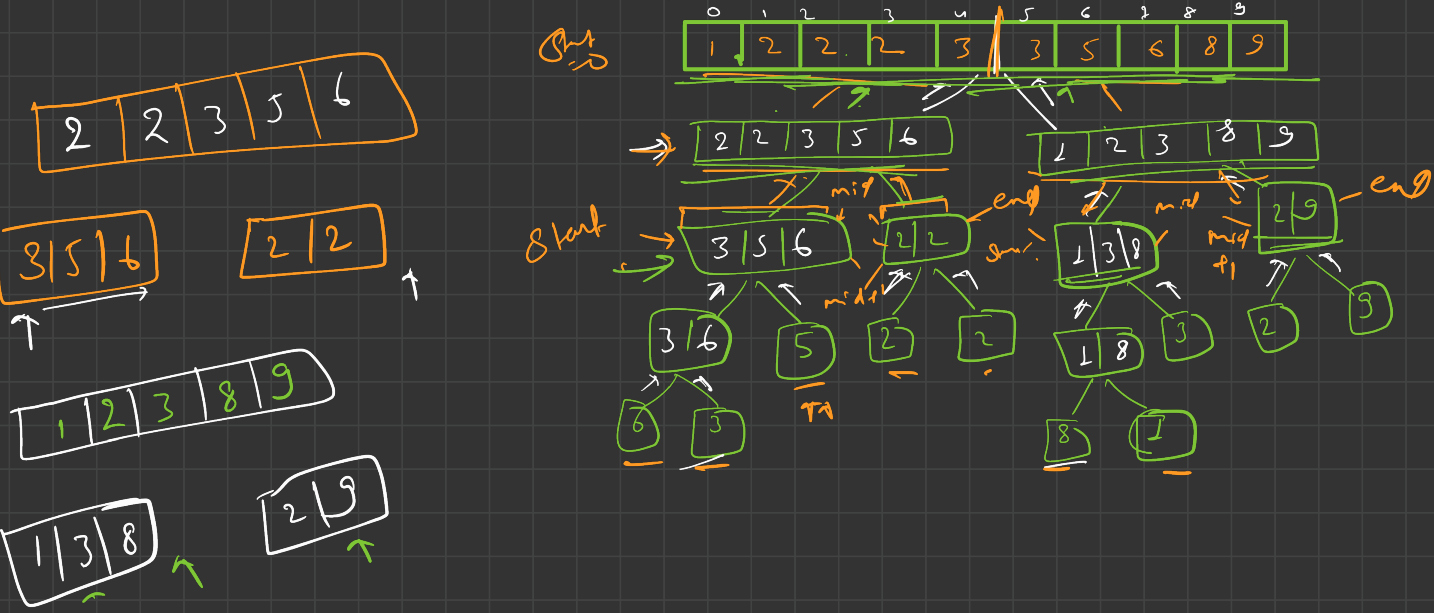
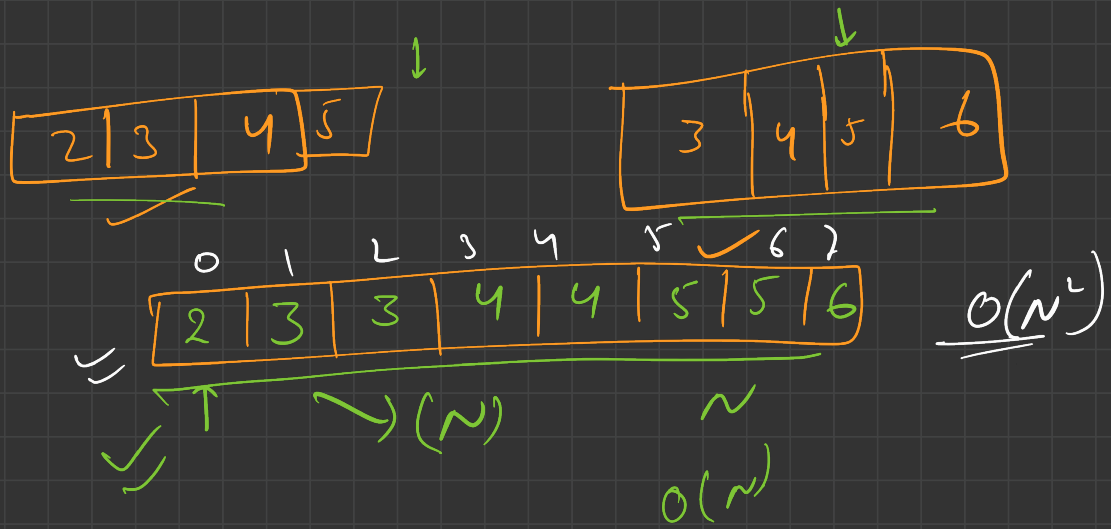
Divide & merge



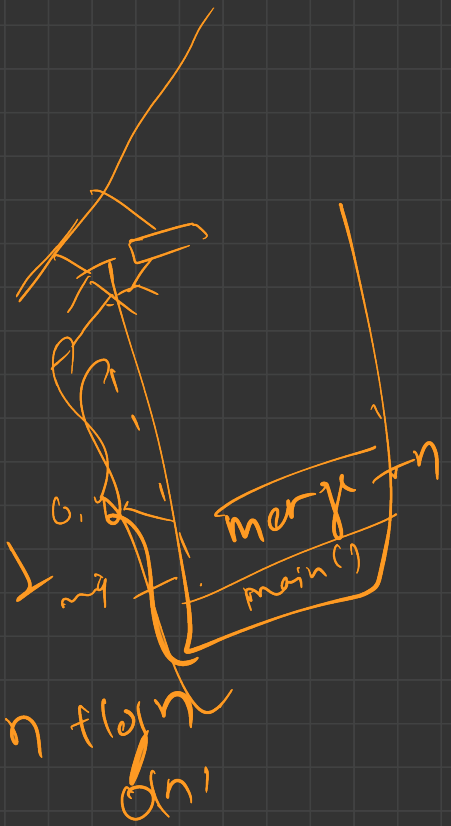




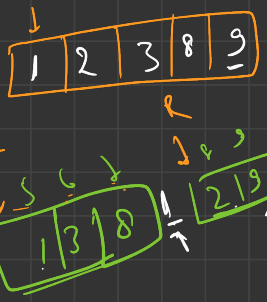
```
void merge sort (int arr[], int start, int end) {
    if (start == end) {
        return;
    }
    int mid = start + (end - start)/2;
    merge sort (arr[], start, mid); // Left
    merge sort (arr[], mid+1, end); // Right
    merge (arr, start, end, mid)
}

int main () {
    merge sort (arr, 0, 9);
}
```

$n+1 \approx n$

$O(n)$



$O(n \log n)$



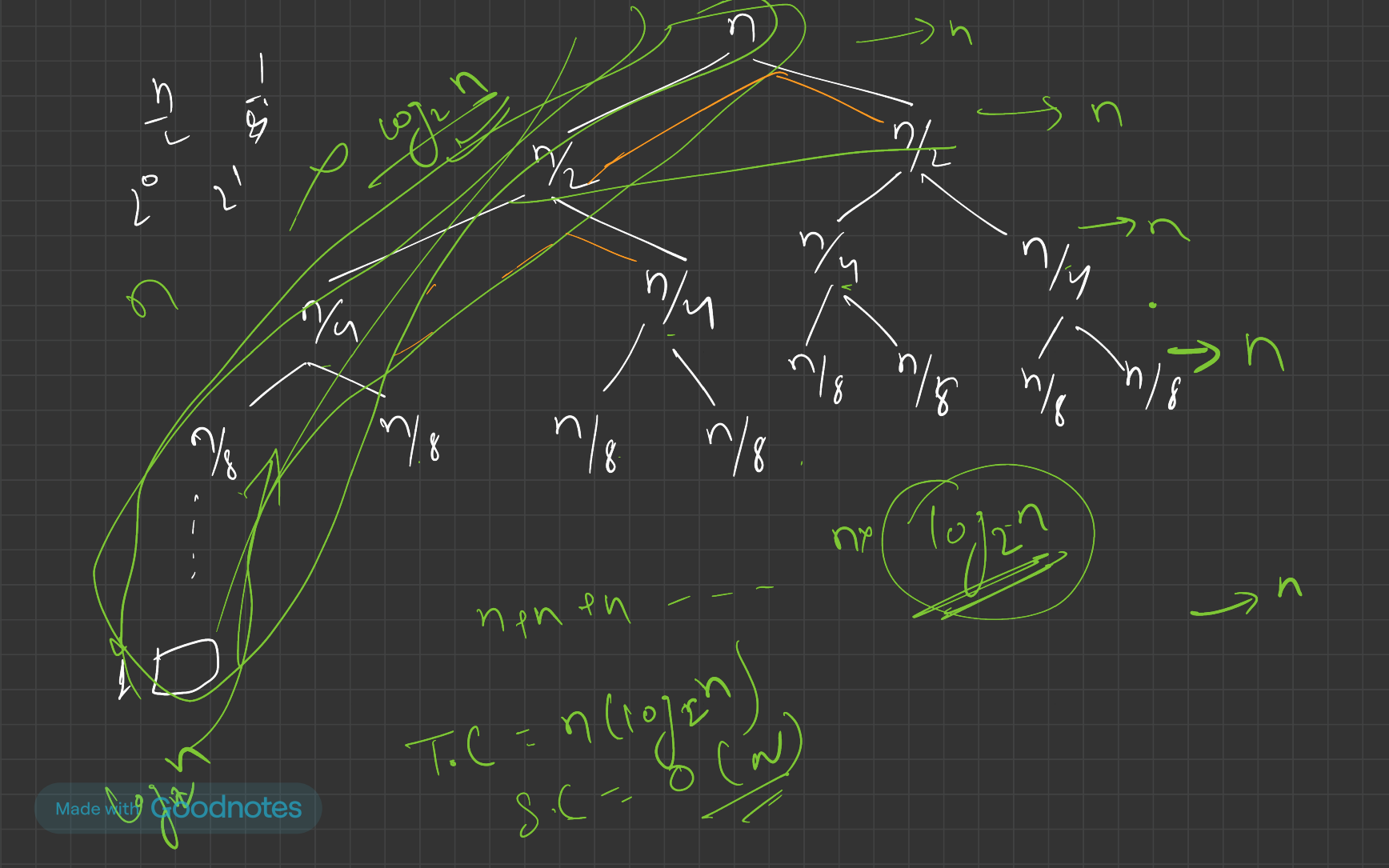
start : 5
mid : 7
end : 9

(end - start) + 1
(9 - 5) + 1
4 + 1
= 5

```
merge ( int arr[], int start, int mid, int end){
    vector<int> temp ((end - start) + 1) ;
    int left = start , Right = mid + 1 , i = 0;
    while (left <= mid && Right <= end) {
        if (arr[left ] <= arr [Right ]) {
            temp[i] = arr [left];
            i++, left++; }
        else {
            temp [i] = arr [Right ];
            i++, Right ++; }
    }
    // left array me bacha hai
    while (left <= mid){
        temp [i] = arr[left]
        i++, left++;
    }
    // right
    while( Right <= end){
        temp[i]= arr[Right];
        i++, Right++;
    }
```

next page is left

```
index : 0
while (start <= end) {
    arr[start] = temp[i];
    start ++, i ++;
} }
```


n
n/2
n/2
n/4
n/4
n/4
n/4
n/8
n/8
n/8
n/8
n/8
n/8
n/8
n/8
n
n
n
n
n
n+n+n ---
n log2n
T.C = n(log2n)
S.C = O(n)
log2n
2^0 2^1 2^3