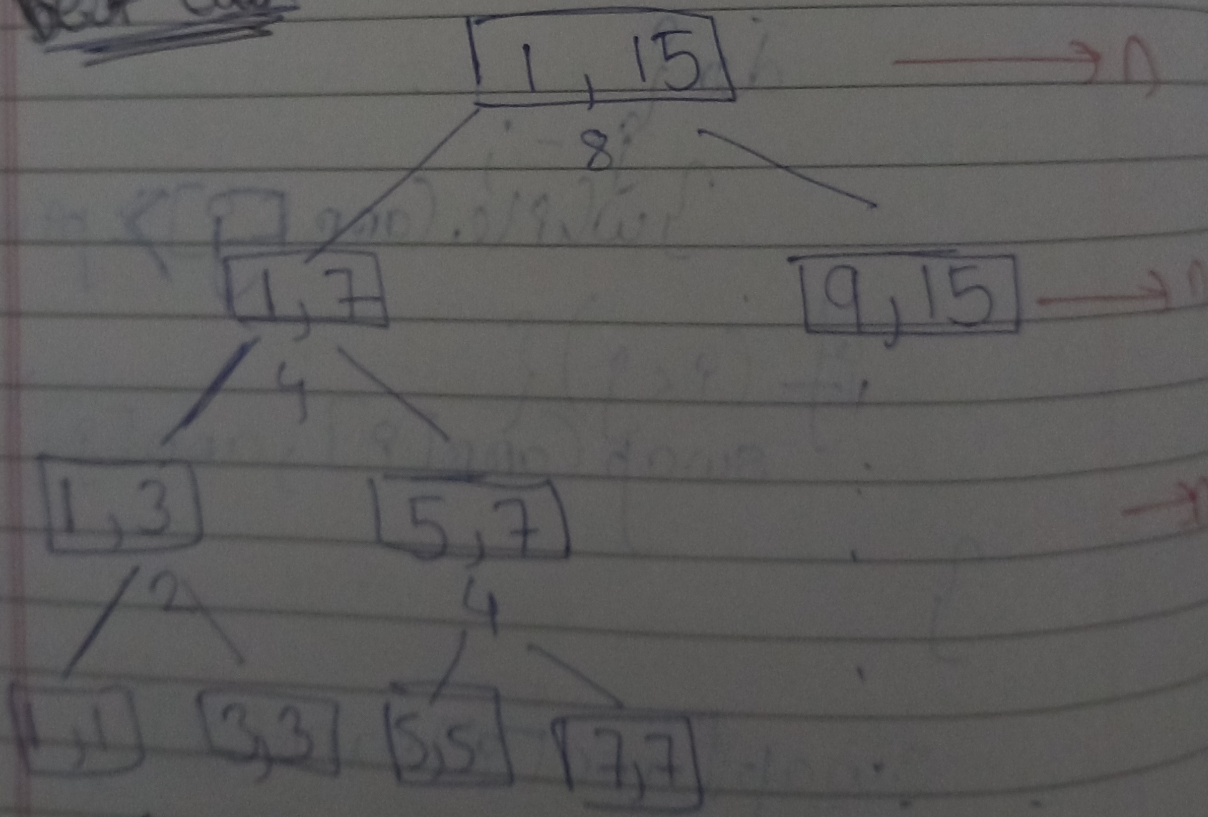
Algorithm :-

```
partition(l, h) {
pivot = arr[l];
    i=l ; j=h;
while(i<j){
do{
    i++;
}while.(arr[i]<= pivot);
do{
    j--;
}while (arr[j] > pivot)
if(i<j){
    swap(arr[i],arr[j]);
}
    swap (arr[j],arr[l]);
    return j;
}
```

```
quickSort (l, h){
  if (l < h){
    j = partition (l, h);
    quickSort (l, p);
    quickSort (j+1, h);
  }
}
```

best case



T.C. = O(n log n) → best case

worst case

↳ already sorted list

eg:- 0 4 8 10 16 18 1

eg:- n ← [1, 7]

n-1 ← [2, 7]

n-2 ← [3, 7]

[4, 7]

[5, 7]

[6, 7]

[7]

⋮

2

1

n(n+1)/2

∴ T.C. = $O(n^2)$

↳ worst case