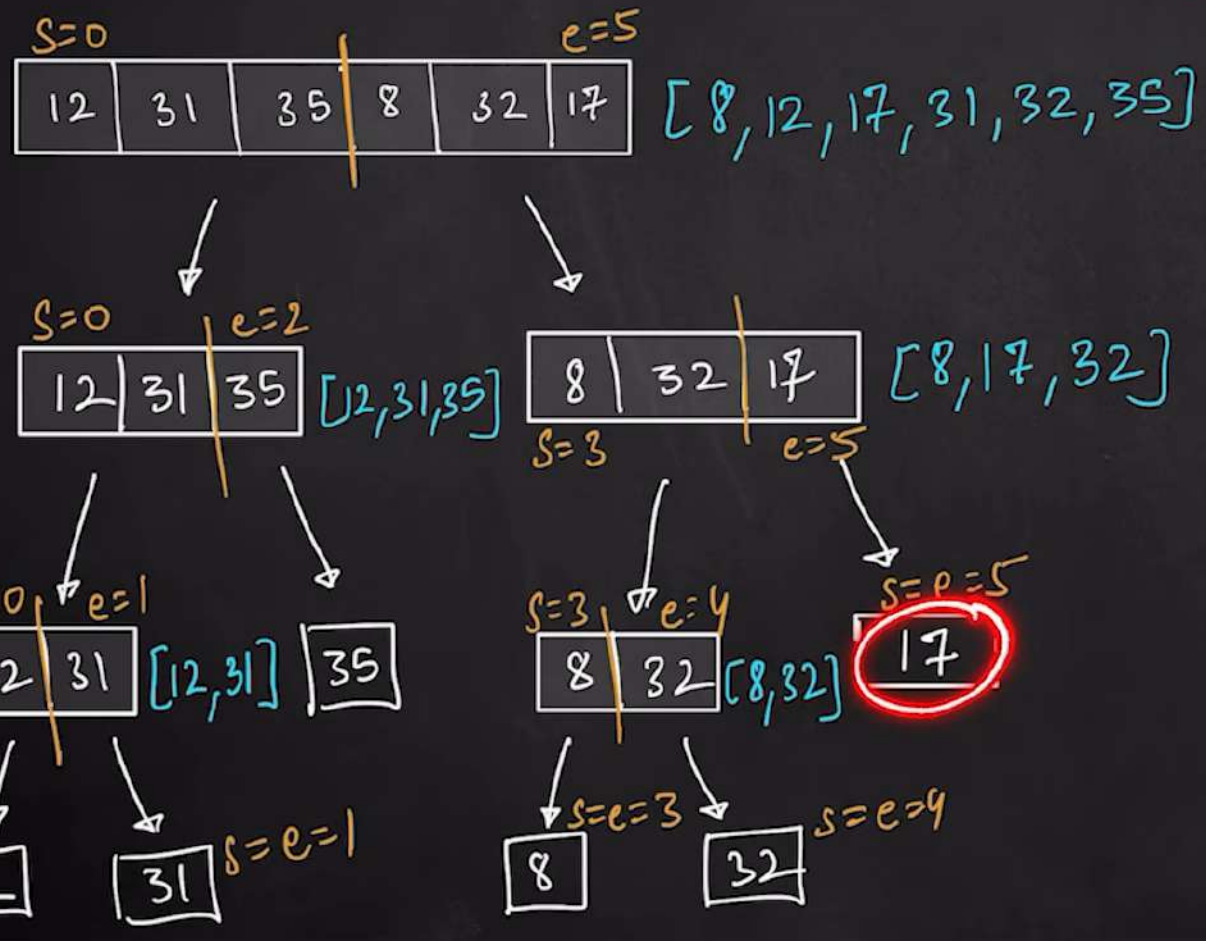
Merge Sort

Divide & Conquer

arr[] = {12, 31, 35, 8, 32, 17}



if(s<e)

s==e

1. Divide the array
   mid
2. merge parts to create a sorted array

1

Merge Sort

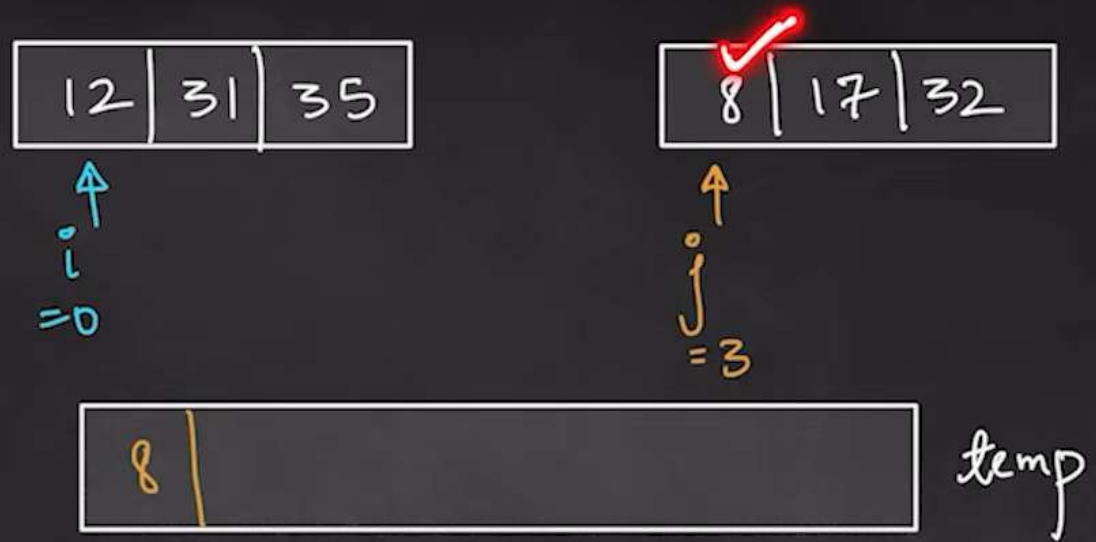
Recursive Function

```
void mergeSort(arr[], start, end) {
  if(s < e) {
    int mid = st + (end-st)/2
    mergeSort(arr, st, mid)   // Left
    mergeSort(arr, mid+1, end) // Right
    merge(arr, st, mid, end)
  }
}
```

Merge Sort

Merge Step

st=0, mid=2, end=5



Merge Sort

Merge Step

st=0, mid=2, end=5

Merge Sort

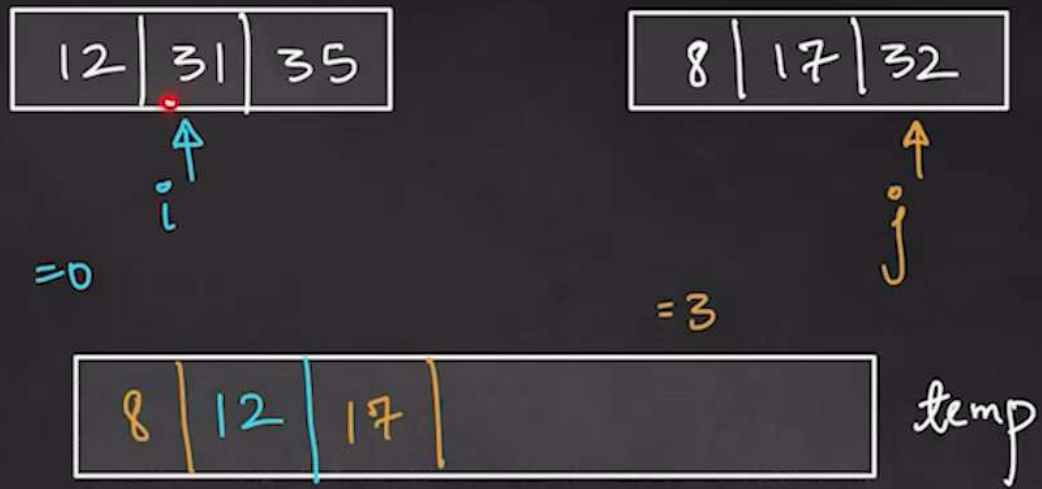
Merge Step

st=0, mid=2, end=5

Merge Sort

Merge Step

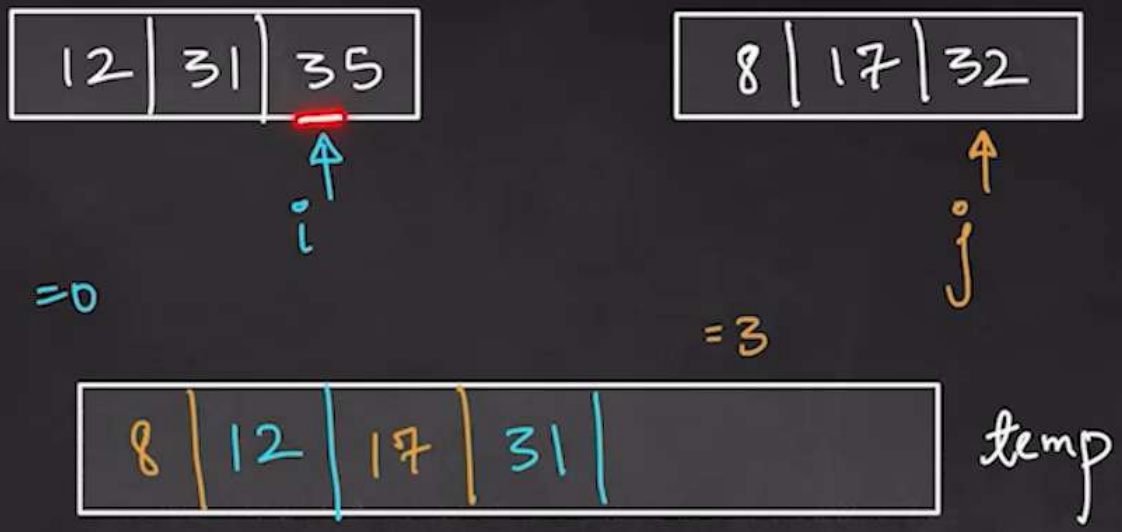
st=0, mid=2, end=5



Merge Sort

Merge Step

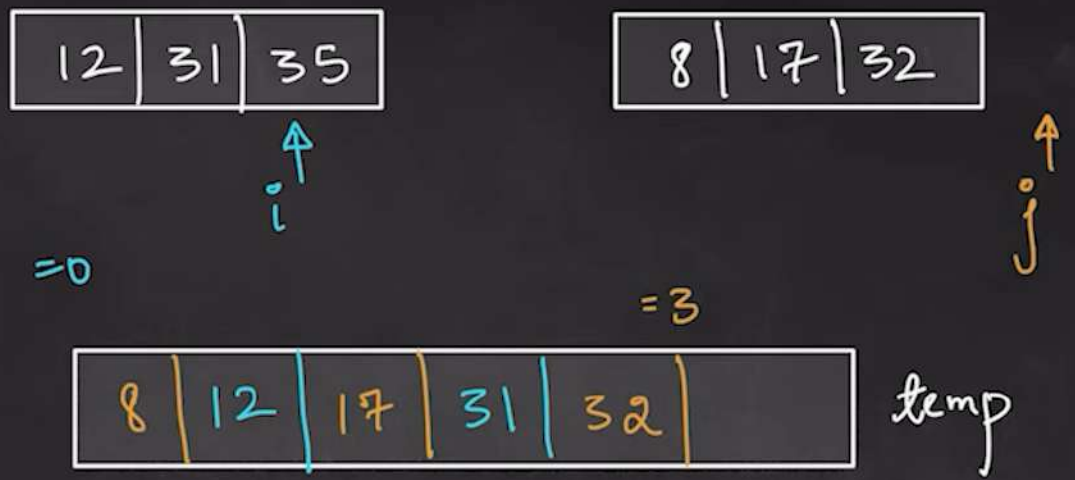
st=0, mid=2, end=5



Merge Sort

Merge Step

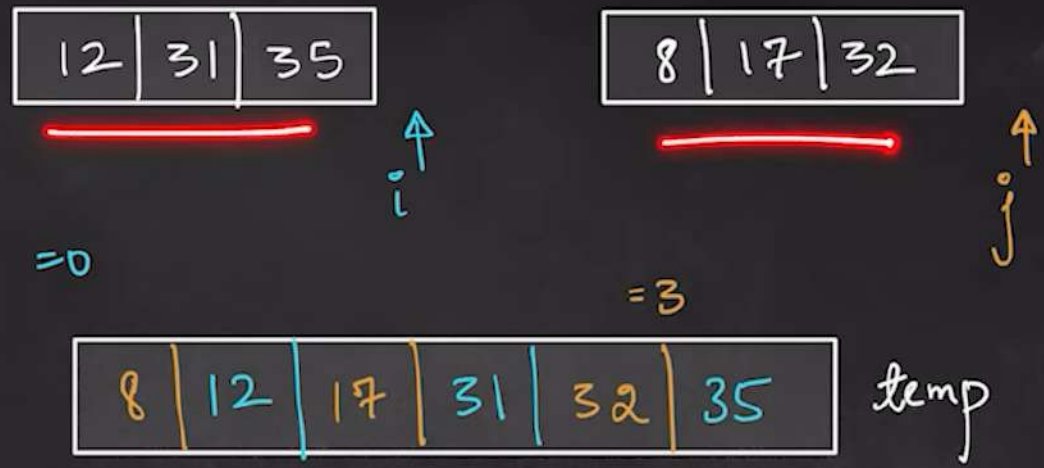
st=0, mid=2, end=5



Merge Sort

Merge Step

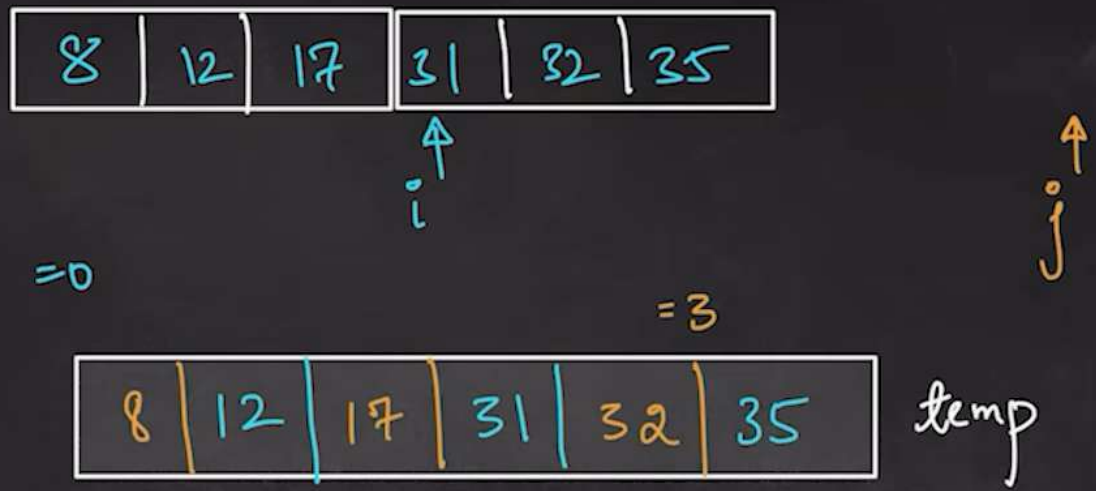
st=0, mid=2, end=5



Merge Sort

Merge Step

st=0, mid=2, end=5



Merge Sort

2

Merge Step

```
void merge(arr, st, mid, end) {
    vector<int> temp;
    i = st, j = mid+1;

    while(i<=mid && j<=end) {
        if(A[i] <= A[j]) {
            temp.pb(A[i])
            i++;
        } else {
            temp.pb(A[j])
            j++
        }
    }
```

Left

```
while(i<=mid) {
    temp.pb(A[i])
    i++
}
```

Right

```
while(j<=end) {
    temp.pb(A[j])
    j++
}
```

```
for(idx=0; idx<temp.size(); idx++)
    A[idx+st] = temp[idx]
```