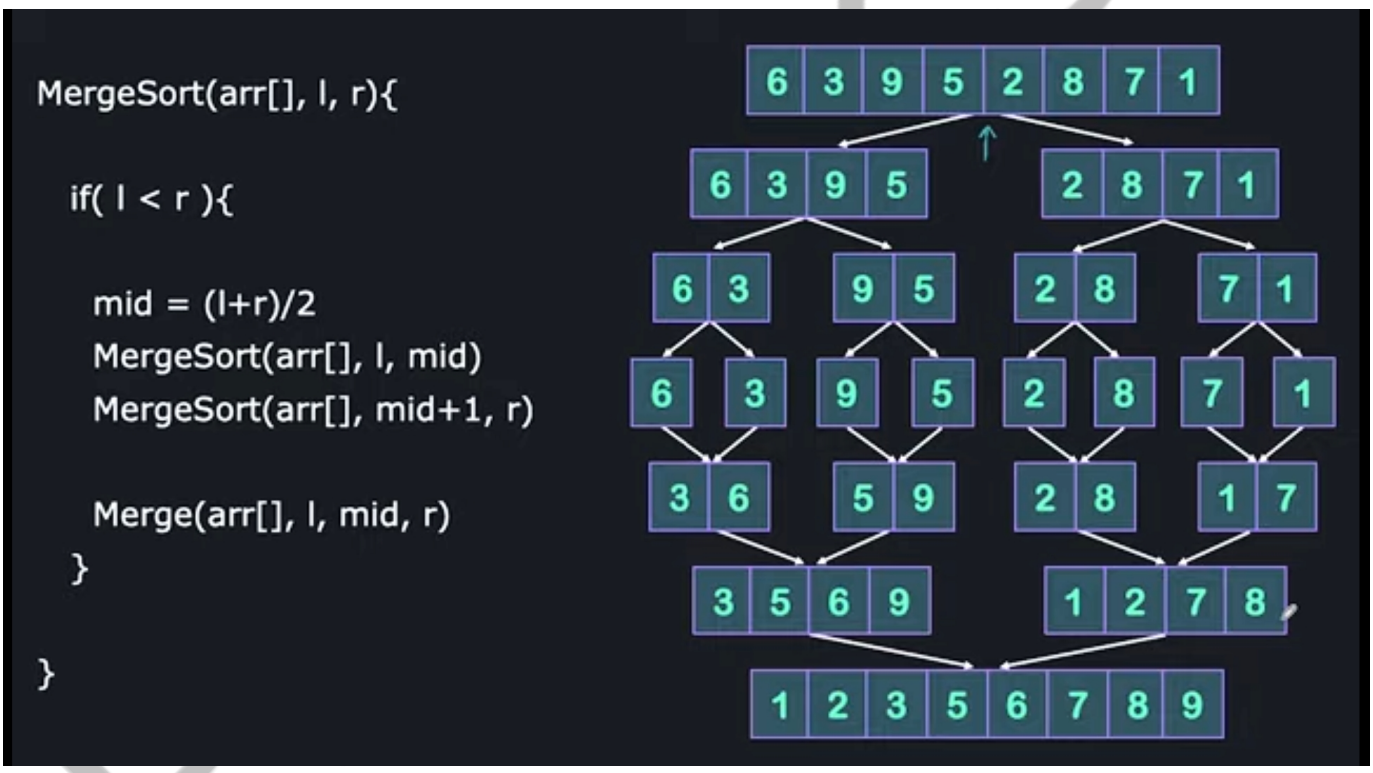
Divide And Conquer

Merge sort

Idea: Divide the array into two parts, sort the left part and sort the right part and merge them

Time Complexity: O(N logN)

Space Complexity: O(N) Since we need an arbitrary array as well.



```
void merge(int arr[], int l, int mid, int r) {
    int n1 = mid - l + 1;
    int n2 = r - mid;
    int a[n1];
    int b[n2]; //temp arrays
    for (int i = 0; i < n1; i++) {
        a[i] = arr[l + i];
    }
    for (int i = 0; i < n2; i++) {
        b[i] = arr[mid + 1 + i];
    }
    int i = 0;
    int j = 0;
    int k = l;
    while (i < n1 && j < n2) {
        if (a[i] < b[j]) {
            arr[k] = a[i];
            k++; i++;
        }
        else {
            arr[k] = b[j];
            k++; j++;
        }
    }
    while (i < n1) {
        arr[k] = a[i];
        k++; i++;
    }
    while (j < n2) {
        arr[k] = b[j];
        k++; j++;
    }
}
```

```
void mergeSort(int arr[], int l, int r) {

    if (l < r) {
        int mid = (l + r) / 2;
        mergeSort(arr, l, mid);
        mergeSort(arr, mid + 1, r);
        merge(arr, l, mid, r);
    }
}
```