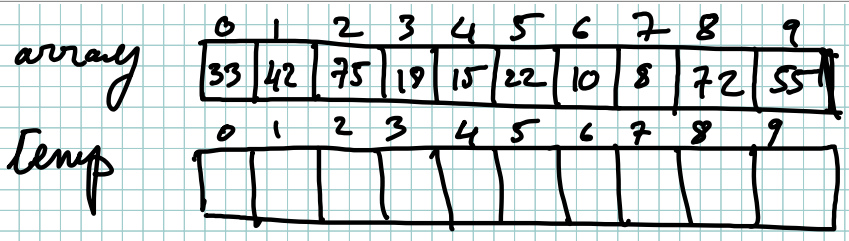
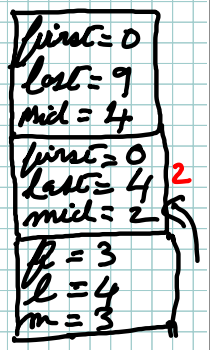
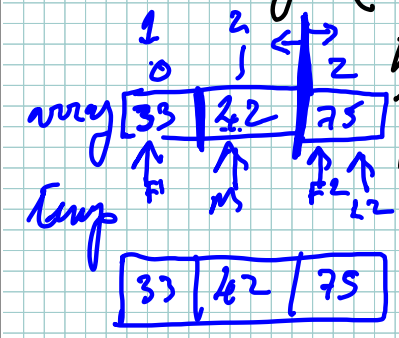





```
mergeSort(array, temp, first, last) {
    if (first < last) {
        mid = (first + last)/2
left    [mergeSort(array, temp, first, mid)
right   [mergeSort(array, temp, mid+1, last)
        merge(array, temp, first, mid, last);
    }
}
```

$O(n \log n) \ll O(n^2)$ (much smaller)

heavy use of storage
double storage for array + temp.




```
Merge(array, temp, first, mid, last){
    first1 = first;   0
    last1 = mid;      0
    first2 = mid+1;   2
    last2 = last;     1
    indexc = first1;  1
    while((first1 <= last1) && (first2 <= last2)){
        if(array[first1] < array[first2]){
            temp[indexc] = array[first1]
            first1++;
        }else{
            temp[indexc] = array[first2]
            first2++
        }
        indexc++
    }
```

```
while (first1 <= last1){
    temp[midsc] = array[first1];
    first1++;
    midsc++;
}
while (first2 <= last2){
    temp[midsc] = array[first2];
    first2++;
    midsc++;
}
for (i = first; i<=last; i++){
    array[i] = temp[i];
}
}end of merge
```