

## Dry-Run

### Merge sort:

int main () {

int arr[] = { 64, 34, 25, 12, 22, 11 }

64	34	25	12	22	11
----	----	----	----	----	----

int n = 6

merge sort ( arr, 0, n-1 )

// mergesort function:

void mergesort ( int arr[], int l, int r )

64	34	25	12	22	11
----	----	----	----	----	----

l —

— r

DATE: \_\_\_\_\_

DAY: \_\_\_\_\_

i)  $(l < r) \rightarrow \text{True}$   $0 < 5$

{

$$\text{int } m = l + (r - l) / 2$$

$$= 1 + (5) / 2$$

$$\text{int } m = 3$$

merge sort (arr, l, m);

merge sort (arr, m+1, r);

64	34	25	12	22	11
----	----	----	----	----	----

$l \quad \quad \quad m \quad \quad \quad m+1 \quad \quad \quad r$

$L[n1] \rightarrow$ 

64	34	25	12
----	----	----	----

 $\leftarrow R[n2]$ 

22	11
----	----

$L[n1]$ 

64	34	25
----	----	----

12
----

 $R[n2]$ 

22
----

 $L[n1]$ 

11
----

 $R[n2]$

$L[n1]$ 

64	34
----	----

25
----

 $R[n2]$

$L[n1]$ 

64
----

34
----

 $R[n2]$

merge ( arr , l, m, r ) ;

64	34	25	12	22	11
----	----	----	----	----	----

$$n1 = 3 - 0 + 1 = 4$$

$$n2 = 5 - 3 = 2$$

// use this process

L[n1] , R[n2]

for breaking whole  
list.

// Merging back.

while i=0 , j=0 , k=0.

while ( <sup>0 < i</sup> i < n1 & & <sup>0 < j</sup> j < n2 ) {

if ( L[i] <= R[j] ) {      64 <= 34

arr[k] = L[i]

i++;

}

else {

arr[k] = R[j]      arr[0] = 34

j++;

j++

}

k++

k++.

}

34	64
----	----



DATE: \_\_\_\_\_

DAY: \_\_\_\_\_

34	64
----	----

25	34	64
----	----	----

12	25	34	64
----	----	----	----

11	22
----	----

11	12	22	25	34	64
----	----	----	----	----	----

Sorted array.