Data Structures and Algorithms
(ES221)

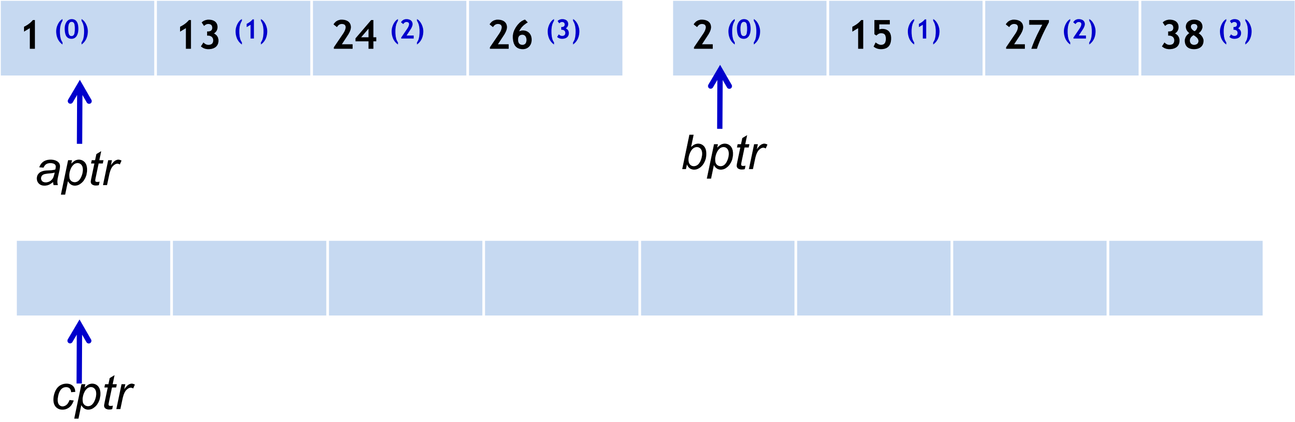
Merge Sort

Dr. Zubair Ahmad

Merge Sort

- The fundamental operation in this algorithm is merging two sorted lists.
- Because the lists are sorted, this can be done in one pass through the input, if the output is put in a third list.
- The basic merging algorithm takes
  - two input arrays: *a* and *b*,
  - an output array: *c*
  - three counters: *aptr*, *bptr*, and *cptr*,
    - which are initially set to the beginning of their respective arrays.
- The smaller of *a*[*aptr*] and *b*[*bptr*] is copied to the next entry in *c*, and the appropriate counters are advanced.
- When either input list is exhausted, the remainder of the other list is copied to *c*.

Merge Sort : Example



1 (0) | 13 (1) | 24 (2) | 26 (3)

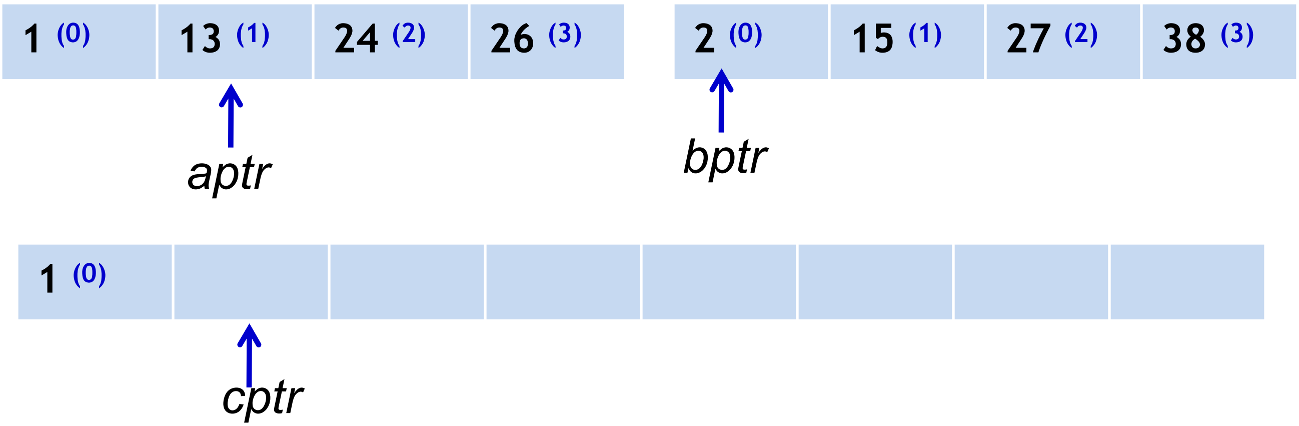
2 (0) | 15 (1) | 27 (2) | 38 (3)

aptr

bptr

cptr

Merge Sort : Example



1 (0)	13 (1)	24 (2)	26 (3)	2 (0)	15 (1)	27 (2)	38 (3)

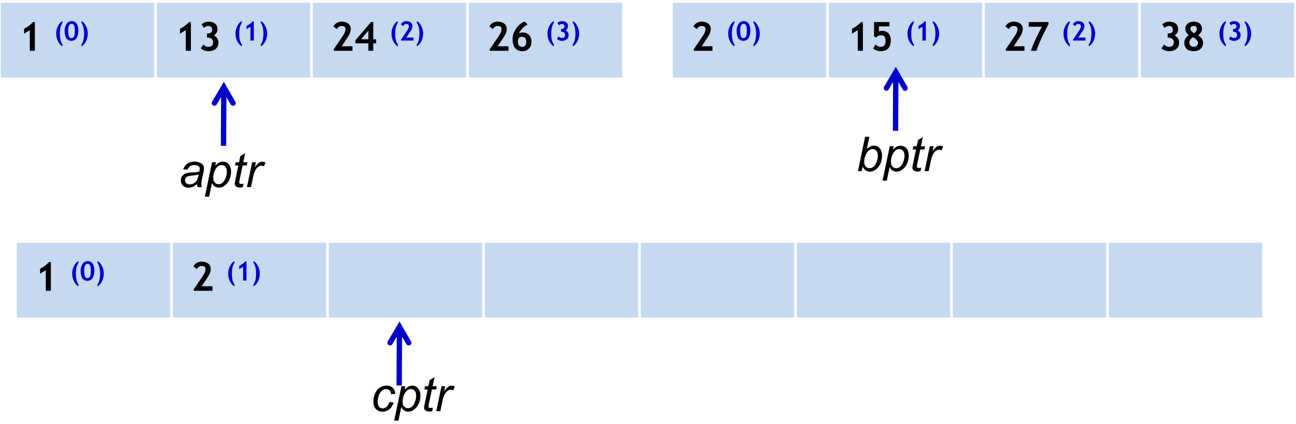
aptr

bptr

1 (0)							

cptr

Merge Sort : Example



1 (0)	13 (1)	24 (2)	26 (3)

2 (0)	15 (1)	27 (2)	38 (3)

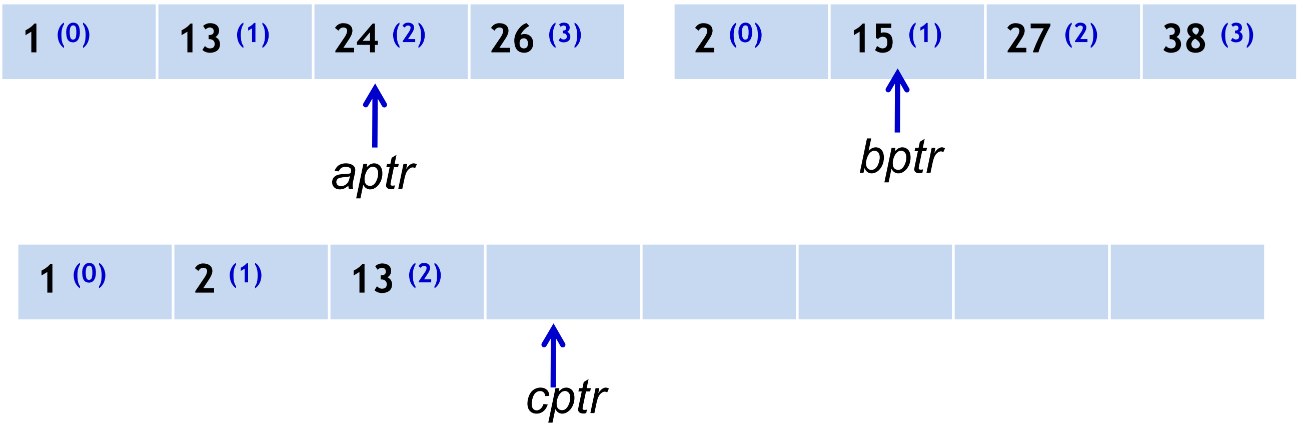
aptr

bptr

1 (0)	2 (1)						

cptr

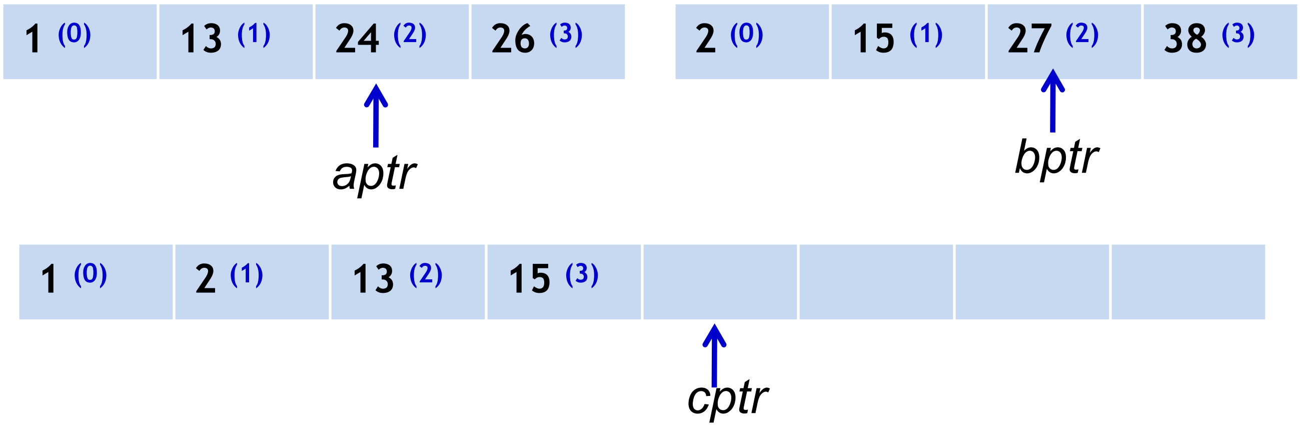
Merge Sort : Example



1 (0)	13 (1)	24 (2)	26 (3)		2 (0)	15 (1)	27 (2)	38 (3)

1 (0)	2 (1)	13 (2)					

Merge Sort : Example

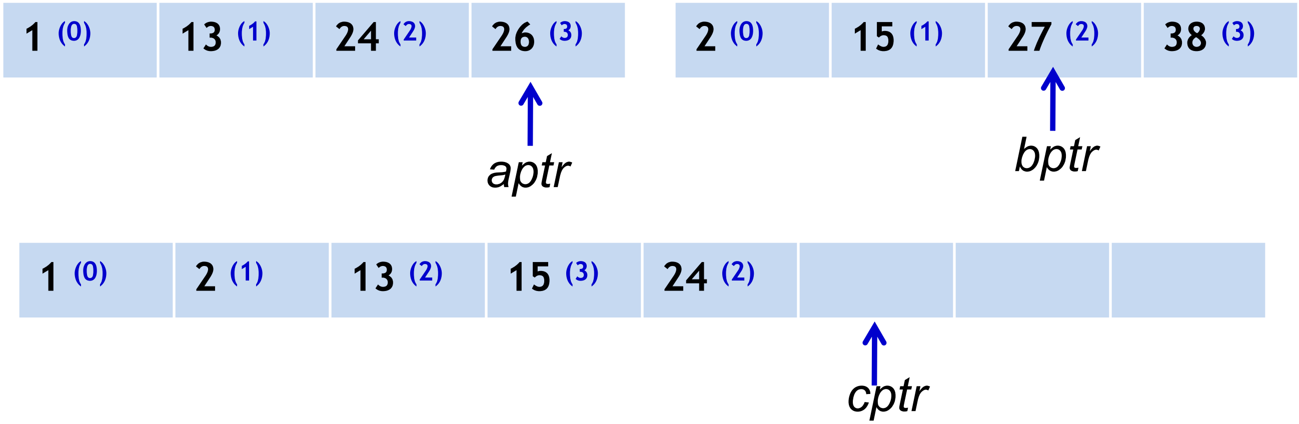


1 (0)	13 (1)	24 (2)	26 (3)

2 (0)	15 (1)	27 (2)	38 (3)

1 (0)	2 (1)	13 (2)	15 (3)				

Merge Sort : Example

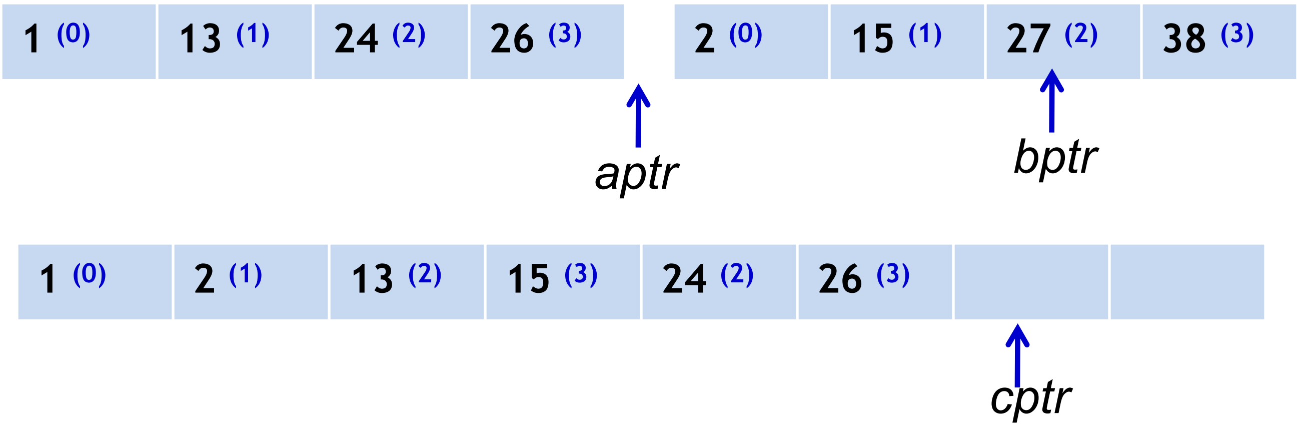


1 (0)	13 (1)	24 (2)	26 (3)

2 (0)	15 (1)	27 (2)	38 (3)

1 (0)	2 (1)	13 (2)	15 (3)	24 (2)			

Merge Sort : Example

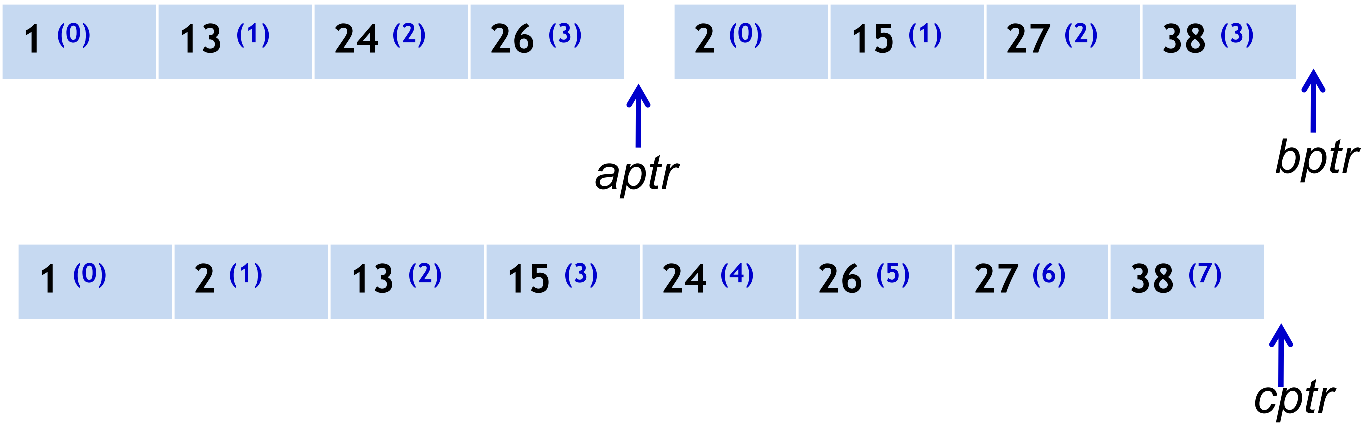


1 (0)	13 (1)	24 (2)	26 (3)

2 (0)	15 (1)	27 (2)	38 (3)

1 (0)	2 (1)	13 (2)	15 (3)	24 (2)	26 (3)		

Merge Sort : Example



1 (0)	13 (1)	24 (2)	26 (3)

2 (0)	15 (1)	27 (2)	38 (3)

1 (0)	2 (1)	13 (2)	15 (3)	24 (4)	26 (5)	27 (6)	38 (7)

Merge Sort

```
void mergesort( input_type a[], unsigned int n )
{
input_type *tmp_array;
tmp_array = new input_type [n];
if( tmp_array != NULL )
   {
   m_sort( a, tmp_array, 0, n-1 );
  delete [](temp_array);
   [illegible]
    }
 else
     cout<<"No space for tmp array!!!”   ;
}
```

Merge Sort

```
void m_sort( input_type a[], input_type tmp_array[ ], int left, int right )
{
int center;
if( left < right )
   {
   center = (left + right) / 2;
   m_sort( a, tmp_array, left, center );
   m_sort( a, tmp_array, center+1, right );
   merge( a, tmp_array, left, center+1, right );
   }
}
```

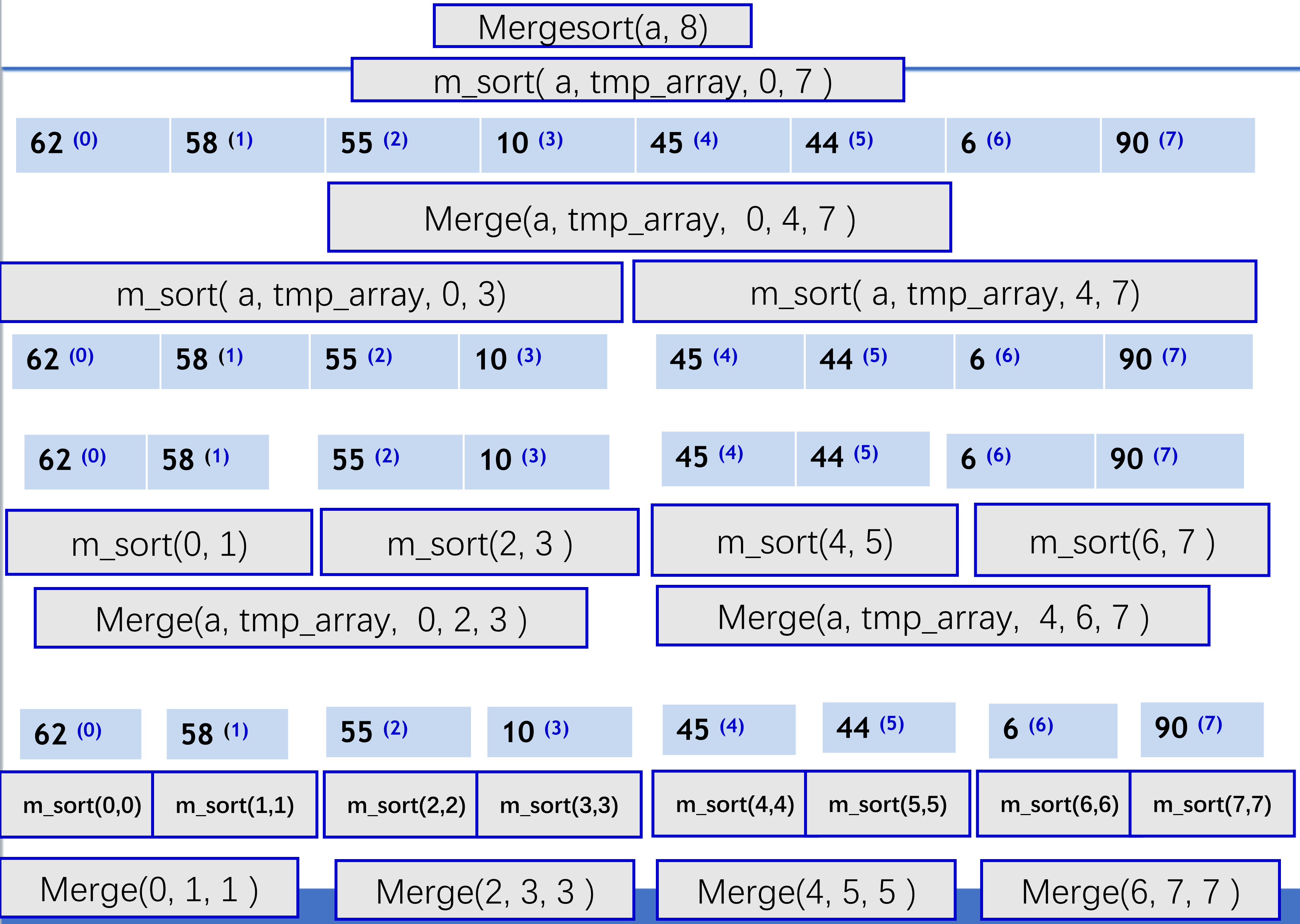
Calculate the **centre** index of the input list

Recursively call the **m_sort** procedure
for t

Recursively call the **m_sort** procedure
for the **right-half** of the input data

Merge the two sorted lists

Merge Sort Example (recursive Function Calls)



Mergesort(a, 8)

m_sort(a, tmp_array, 0, 7)

62 (0) 58 (1) 55 (2) 10 (3) 45 (4) 44 (5) 6 (6) 90 (7)

Merge(a, tmp_array, 0, 4, 7)

m_sort(a, tmp_array, 0, 3)

m_sort(a, tmp_array, 4, 7)

62 (0) 58 (1) 55 (2) 10 (3)

45 (4) 44 (5) 6 (6) 90 (7)

62 (0) 58 (1)

55 (2) 10 (3)

45 (4) 44 (5)

6 (6) 90 (7)

m_sort(0, 1)

m_sort(2, 3)

m_sort(4, 5)

m_sort(6, 7)

Merge(a, tmp_array, 0, 2, 3)

Merge(a, tmp_array, 4, 6, 7)

62 (0)

58 (1)

55 (2)

10 (3)

45 (4)

44 (5)

6 (6)

90 (7)

m_sort(0,0) m_sort(1,1)

m_sort(2,2) m_sort(3,3)

m_sort(4,4) m_sort(5,5)

m_sort(6,6) m_sort(7,7)

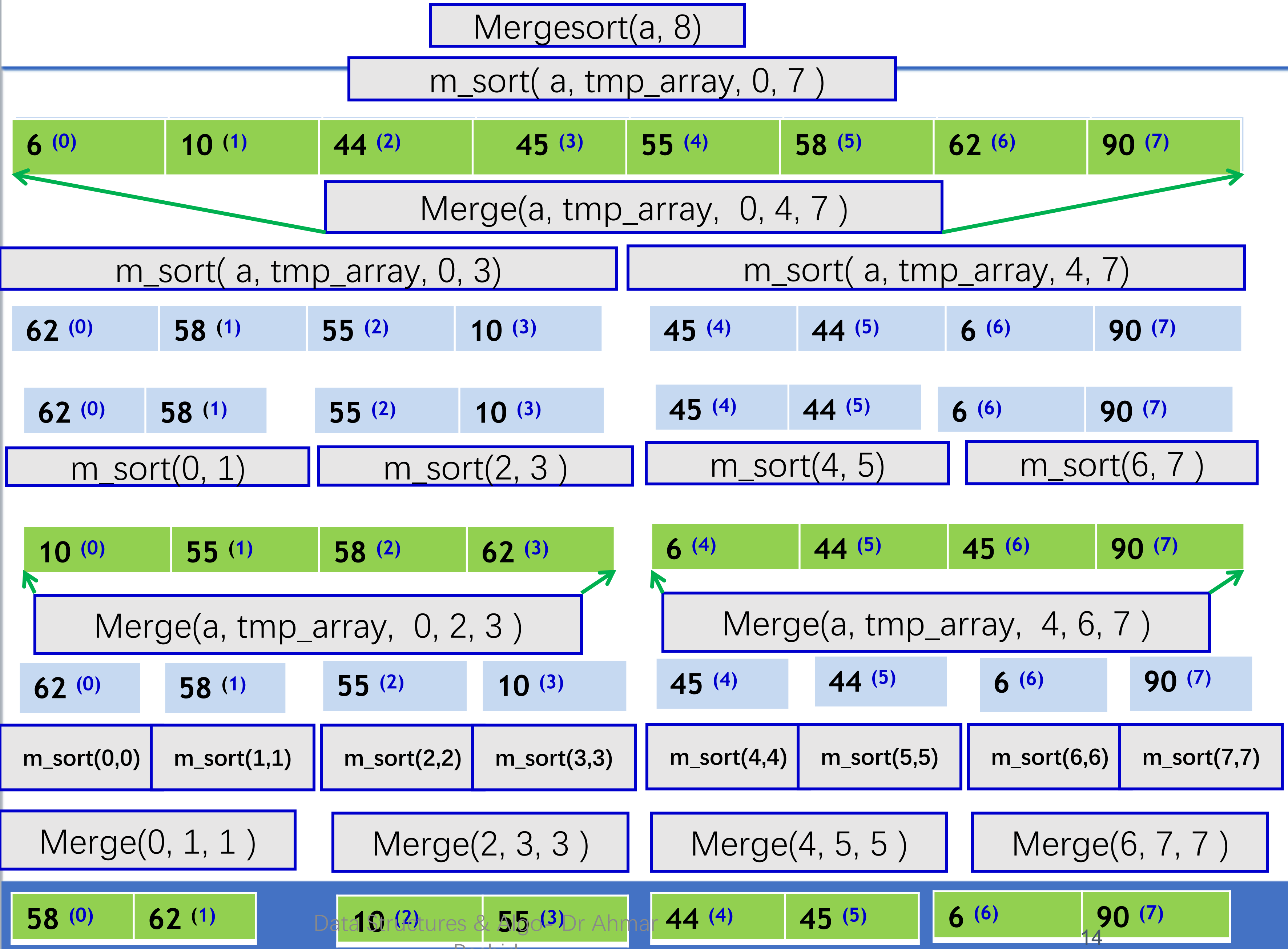
Merge(0, 1, 1)

Merge(2, 3, 3)

Merge(4, 5, 5)

Merge(6, 7, 7)

Merge Sort Example (Merging process)



Mergesort(a, 8)

m_sort(a, tmp_array, 0, 7)

6 (0) | 10 (1) | 44 (2) | 45 (3) | 55 (4) | 58 (5) | 62 (6) | 90 (7)

Merge(a, tmp_array, 0, 4, 7)

m_sort(a, tmp_array, 0, 3)

m_sort(a, tmp_array, 4, 7)

62 (0) | 58 (1) | 55 (2) | 10 (3) | 45 (4) | 44 (5) | 6 (6) | 90 (7)

62 (0) | 58 (1) | 55 (2) | 10 (3) | 45 (4) | 44 (5) | 6 (6) | 90 (7)

m_sort(0, 1)

m_sort(2, 3)

m_sort(4, 5)

m_sort(6, 7)

10 (0) | 55 (1) | 58 (2) | 62 (3) | 6 (4) | 44 (5) | 45 (6) | 90 (7)

Merge(a, tmp_array, 0, 2, 3)

Merge(a, tmp_array, 4, 6, 7)

62 (0) | 58 (1) | 55 (2) | 10 (3) | 45 (4) | 44 (5) | 6 (6) | 90 (7)

m_sort(0,0) | m_sort(1,1) | m_sort(2,2) | m_sort(3,3) | m_sort(4,4) | m_sort(5,5) | m_sort(6,6) | m_sort(7,7)

Merge(0, 1, 1)

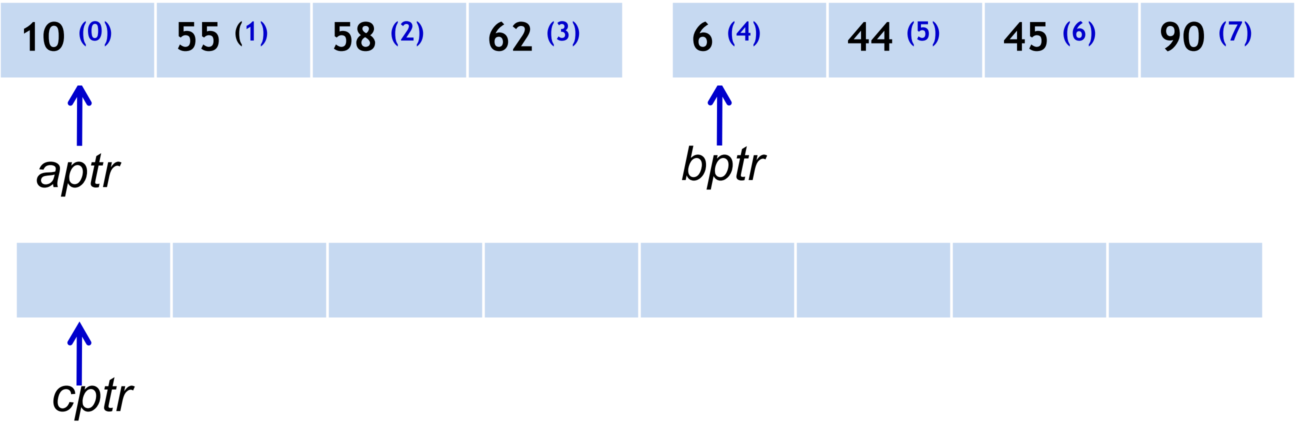
Merge(2, 3, 3)

Merge(4, 5, 5)

Merge(6, 7, 7)

58 (0) | 62 (1) | 10 (2) | 55 (3) | 44 (4) | 45 (5) | 6 (6) | 90 (7)

Merge Sort : Example



10 (0)	55 (1)	58 (2)	62 (3)	6 (4)	44 (5)	45 (6)	90 (7)

aptr

bptr

cptr

62 (0)	58 (1)	55 (2)	10 (3)	45 (4)	44 (5)	6 (6)	90 (7)

Questions?

zahmaad.github.io