#### COMP2521 23T3

wethou

Merging

Implementa

CIOII

Dronarties

Sorting Lists

## COMP2521 23T3 Sorting Algorithms (II)

Kevin Luxa cs2521@cse.unsw.edu.au

merge sort

## Divide-and-Conquer Algorithms

Culibbin

Merging

Implement

Analysi

Properties

Sorting Lists

Bottom-Un

Appendix

divide-and-conquer algorithms
split a problem into smaller sub-problems,
solve the sub-problems recursively,
and then combine the results.

#### Method

Splitting

Merging

Implemen

Analysis

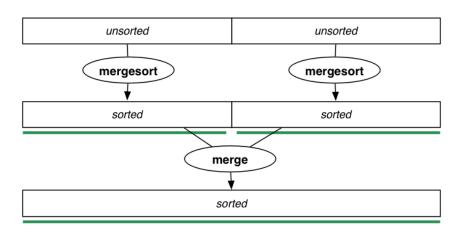
Propertie

Sorting Lists

Bottom-U

A divide-and-conquer sorting algorithm:

split the array into two roughly equal-sized parts.
recursively sort each of the partitions.
merge the two now-sorted partitions into a sorted array.



#### Method

Splitting

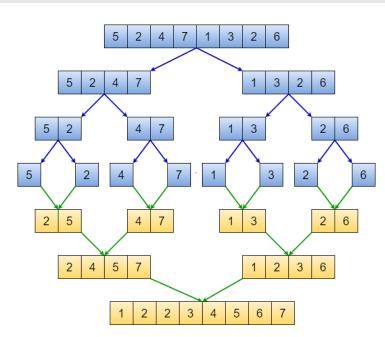
Merging

Impleme

Analysi:

Propertie:

Sorting Lists



# Merge Sort Splitting

Metno

Splitting

Merging

tion

Dronartic

Sorting List

00111115 2101

Bottom-up

How do we split the array?

- We don't physically split the array
- We simply calculate the midpoint of the array

- Then recursively sort each half by passing in appropriate indices
  - Sort between indices lo and mid
  - Sort between indices mid + 1 and hi
- $\bullet$  This means the time complexity of splitting the array is  ${\it O}(1)$

## Merge Sort Merging

Snlittin

Merging Example 1

Example 2 Analysis

Analye

Propertie

Sorting Lists

Bottom-IIn

Appendix

How do we merge two sorted subarrays?

- We merge the subarrays into a temporary array
- Keep track of the smallest element that has not been merged in each subarray
- Copy the smaller of the two elements into the temporary array
  - If the elements are equal, take from the left subarray
- Repeat until all elements have been merged
- Then copy from the temporary array back to the original array

Method

Splittin

Merging

Example 1

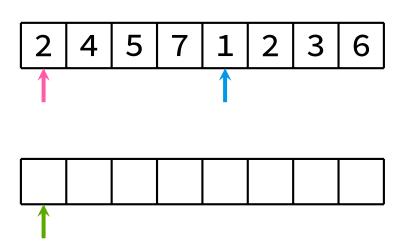
Analysis

Analysis

....

Analysis

JUI LING LISES



Method

.

Merging Example 1

Example

Analysis

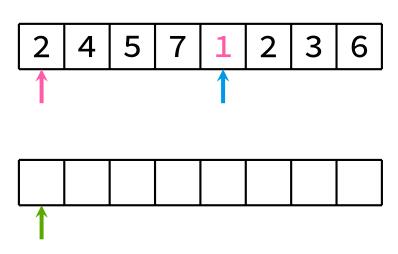
tion

Analysis

**Properties** 

Sorting Lists

Bottom-U



Method

Splittin

Merging

Example 1

Analysis

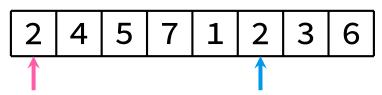
Implemen

Δnalvsi

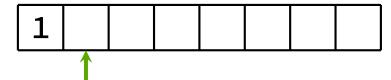
Propertie

Sorting Lists

Appendix



When items are equal, merge takes from the left subarray (this ensures stability)



Mediod

Splittin

Merging

Example 1

Analysis

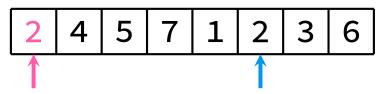
Impleme

Analysi

Propertie

Sorting Lists

Appendix



When items are equal, merge takes from the left subarray (this ensures stability)



Method

Splitting

Merging

Example 1

Analysis

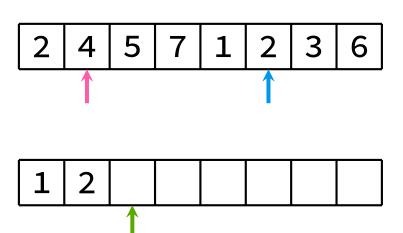
Luculous

Analysis

.

Sorting Lists

Bottom-U



Method

Splitting

Merging

Example 1

Analysis

Analysis

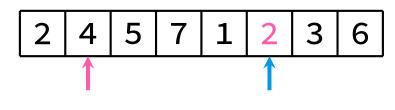
LIOII

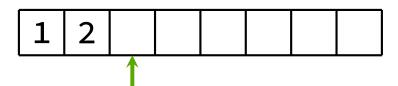
Analysis

.

JUI LING LISES

50000111 0





Method

Splitting

Merging

Example 1

Analysis

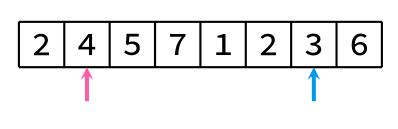
Imnlem

Analysis

7 11 10 Cy 5 1

. .

Ü





Method

Splitting

Merging

Example 1

Analysis

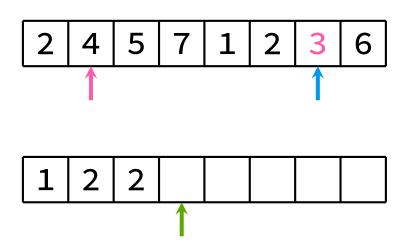
Implon

.....

Analysis

Sorting Lists

Bottom-U



Method

Splitting

Merging

Example 1

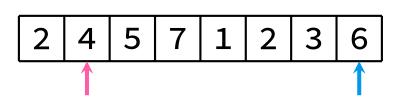
Analysis

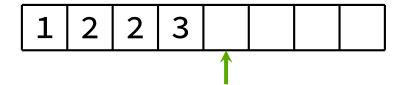
Allatysis

LIOII

Analysis

Sorting Lists





Method

Splitting

Merging

Example 1

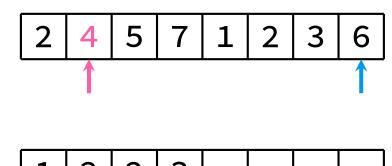
Analysis

Implem

Analysis

. . .

00111115 2101



Method

Splitting

Merging

Example 1

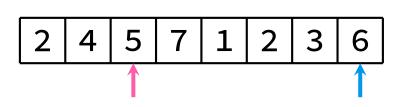
Analysis

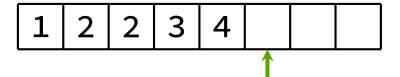
. . .

.....

Analysis

Datham III





Method

Splittin

Merging

Example 1

Analysis

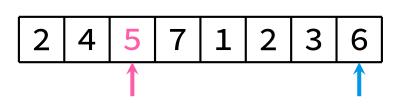
Implen

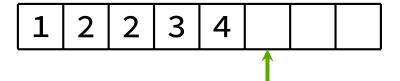
Analysis

Propertie

Sorting Lists

Bottom-U





Method

Splitting

Merging

Example 1

Analysis

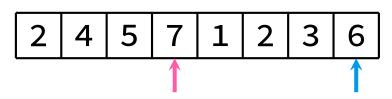
tion

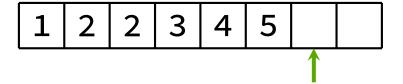
Analysis

· · opercies

Surting List

Annendix





Method

Splittin

Merging

Example 1

Analysis

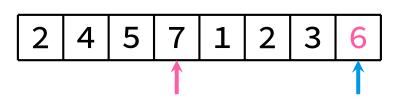
Imnler

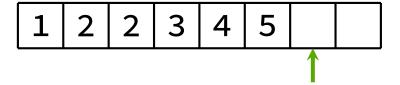
Analysis

7 11 10 Cy 5 1

. . . . . . . . .

Rottom-II





Method

Splitting

Merging

Example 1

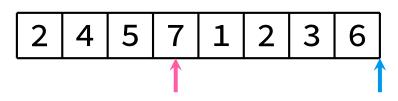
Analysis

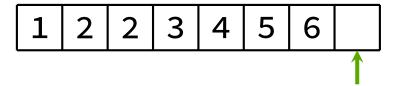
Luculou

----

Analysis

Jui tilig List





Method

Splitting

Merging

Example 1

Analysis

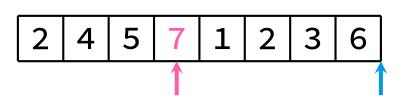
Implom

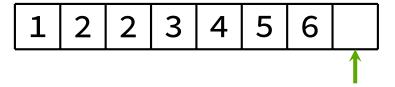
Analysis

Droport

Sorting Lists

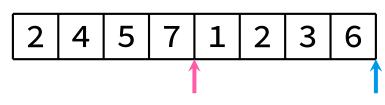
Bottom-U



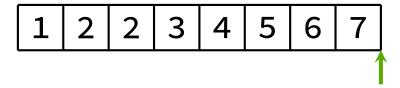


Merging

Example 1



Now copy back to original array



Method

Splitting

Merging Example 1

Example 2

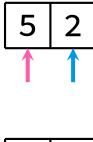
Implementa

Analysis

Properties

Sorting Lists

Bottom-Up



Method

Splitting

Merging Example 1

Example 2

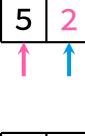
Implementa

Analysis

Properties

Sorting Lists

Bottom-Up





Method

Merging

Example 2

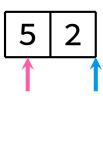
Implements

Analysis

Dronarties

Sorting Lists

Bottom-Up





G 11...1

Merging Example 1

Example 2

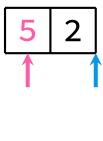
Implementa

Analysis

**Properties** 

Sorting Lists

Bottom-Up



Methou

Merging

Example 2

Implementa

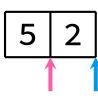
Analysis

Properties

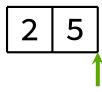
Sorting Lists

Bottom-Up

Appendix



Now copy back to original array



## Merge Sort Merging

Analysis

• The time complexity of merging two sorted subarrays is O(n), where n is the total number of elements in both subarrays

- Therefore:
  - Merging two subarrays of size 1 takes 2 "steps"
  - Merging two subarrays of size 2 takes 4 "steps"
  - Merging two subarrays of size 4 takes 8 "steps"

Method

Mergin

Implementation

Properties

Sorting Lists

Bottom-Up

```
if (lo >= hi) return;
int mid = (lo + hi) / 2;
mergeSort(items, lo, mid);
mergeSort(items, mid + 1, hi);
merge(items, lo, mid, hi);
```

void mergeSort(Item items[], int lo, int hi) {

```
COMP2521
23T3
```

# Merge Sort C Implementation: Merge

```
Merging
Implementa-
```

tion

```
Analysis
Properties
```

```
Sorting Lists
Bottom-Up
```

```
Appendix
```

```
int i = lo, j = mid + 1, k = 0;
// Scan both segments, copying to `tmp'.
while (i <= mid && j <= hi) {</pre>
    if (le(items[i], items[j])) {
        tmp[k++] = items[i++];
    } else {
        tmp[k++] = items[i++];
    }
// Copy items from unfinished segment.
while (i <= mid) tmp[k++] = items[i++];</pre>
while (i <= hi) tmp[k++] = items[i++];</pre>
// Copy `tmp' back to main array.
for (i = lo, k = 0; i <= hi; i++, k++) {
    items[i] = tmp[k];
}
free(tmp):
```

void merge(Item items[], int lo, int mid, int hi) {
 Item \*tmp = malloc((hi - lo + 1) \* sizeof(Item));

## Merge Sort Analysis

Meruoa

Merging

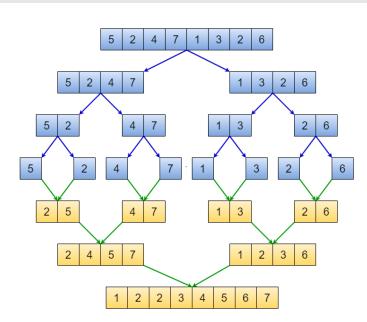
Implement

### Analysis

Propertie:

Sorting Lists

Bottom-Up



## Merge Sort Analysis



Merging

Luculou

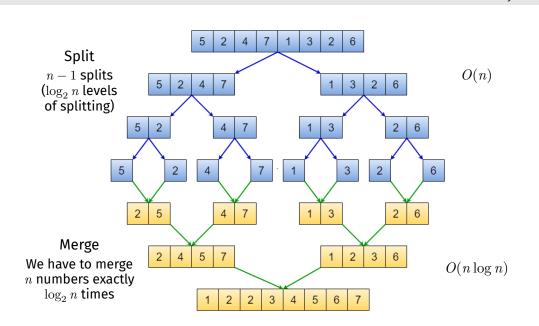
tion

### Analysis

Propertie

Sorting List

Bottom-Up



## Merge Sort Analysis

Methou

Mergin

Implemention

## Analysis

Propertie

Sorting List

Annondiv

### **Analysis:**

- Merge sort splits the array into equal-sized partitions halving at each level  $\Rightarrow \log_2 n$  levels
- The same operations happen at every recursive level
- Each 'level' requires  $\leq n$  comparisons

### Therefore:

- The time complexity of merge sort is  $O(n \log n)$ 
  - Best-case, average-case, and worst-case time complexities are all the same

Mediloc

Mergins

Implement

Analysis

ropercies

JUILING LISES

A 15

Note: Not required knowledge in COMP2521!

Let T(n) be the time taken to sort n elements.

Splitting arrays into two halves takes constant time. Merging two sorted arrays takes n steps.

So we have that:

$$T(n) = 2T(n/2) + n$$

Then the Master Theorem (see COMP3121) can be used to show that the time complexity is  $O(n \log n)$ .

COMP2521 23T3

# Merge Sort Properties

Culitation

Merging

Implemention

Analysi

Properties

Sorting Lists

Bottom-Up

Appendix

### Stable

Due to taking from left subarray if items are equal during merge

### **Non-adaptive**

 $O(n \log n)$  best case, average case, worst case

### Not in-place

Merge uses a temporary array of size up to nNote: Merge sort also uses  $O(\log n)$  stack space

#### Merge Sort on Lists

Splitting

Merging

tion

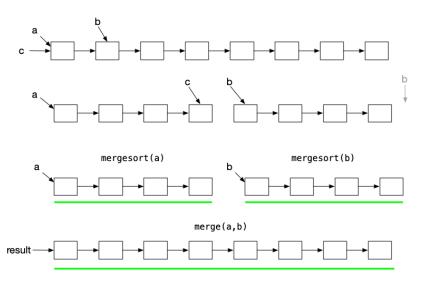
...,..

Порегись

Sorting Lists

Bottom-ob

It is possible to apply merge sort on linked lists.



Method

Splittir

Merging

Implemen

Analysi

Properties

Sorting Lists

Bottom-Up
Implementation

implementatio

An approach that works non-recursively!

- on each pass, our array contains sorted *runs* of length m.
- initially, *n* sorted runs of length 1.
- The first pass merges adjacent elements into runs of length 2.
- The second pass merges adjacent elements into runs of length 4.
- ullet ... continue until we have a single sorted run of length n.

Can be used for external sorting; e.g., sorting disk-file contents

COMP2521 Bottom-Up Merge Sort 23T3 Example [0] [1] [2] [15] Merging Original Ε Ε О After 1st pass sorted slices of length 2 Bottom-Up After 2nd pass Ε R sorted slices of length 4 After 3rd pass sorted slices of length 8 After 4th pass sorted slice of length 16

### Bottom-Up Merge Sort C Implementation

```
Splittin
```

Implom

Analysi

Propertie

Sorting Lists

Implementation

Implementation

```
#define MIN(a, b) ((a) < (b) ? (a) : (b))
void mergeSortBottomUp(Item items[], int lo, int hi) {
    for (int m = 1; m <= lo - hi; m *= 2) {
        for (int i = lo; i <= hi - m; i += 2 * m) {
            int end = MIN(i + 2 * m - 1, hi);
            merge(items, i, i + m - 1, end);
```

Metho

Splittir

Merging

Impleme

Analys

Propertie

Sorting Lists

Implementation

Appendix

https://forms.office.com/r/aPF09YHZ3X



#### COMP2521 23T3

Method

Splittin

Merging

tion

Analysis

Properties

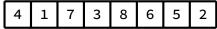
Sorting Li

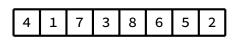
Rottom-IIn

#### Appendix

Merge Sort Demo

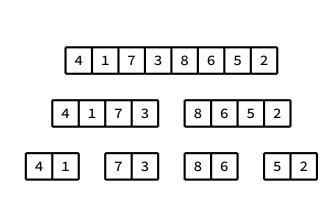
### **Appendix**





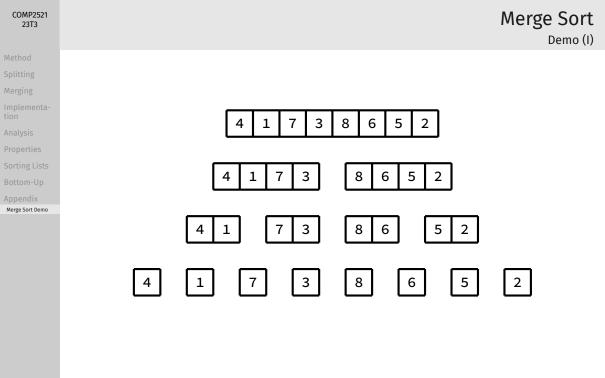
4 1 7 3 8 6 5 2



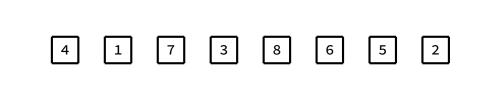


Merge Sort

Demo (I)







Merge Sort

Demo (II)

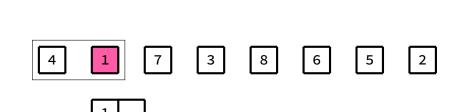


4 1 7 3 8 6 5 2

Merge Sort

Demo (II)

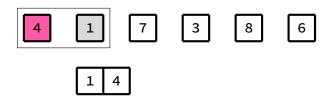


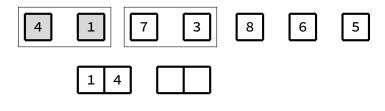


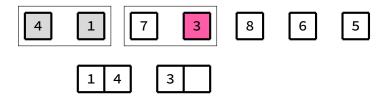
Merge Sort

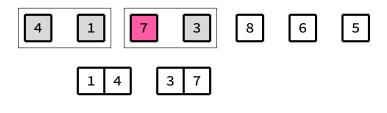
Demo (II)

Merge Sort Demo (II)





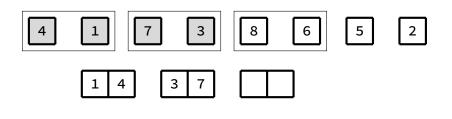


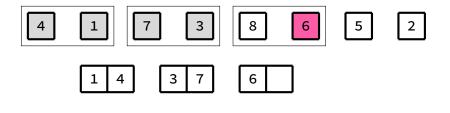




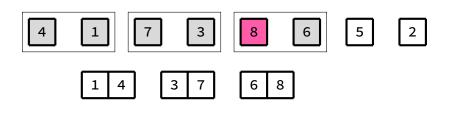
Demo (II)

Merge Sort

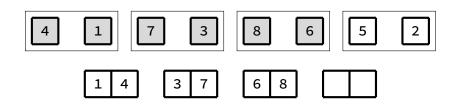




### Merge Sort Demo (II)



### Merge Sort Demo (II)



Splitting

Merging

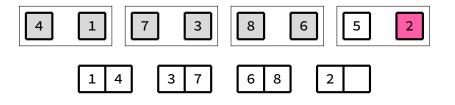
tion

Analysis

Sorting List

3011115 21303

Appendix



Splitting

Merging

tion

Analysi

Troperties

Sorting Lists

Appendix

Appendix Merge Sort Demo 1

8

6

5

TE

2 5

Splitting

Merging

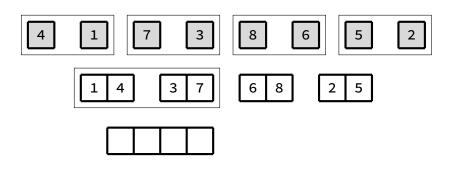
Analysis

Propertie:

Sorting Lists

Bottom-nb

Appendix Merge Sort Dem



Splitting

Merging

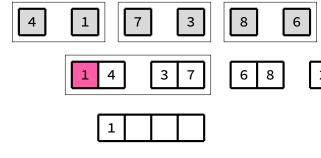
tion

Anatysis

Corting Lie

3011115 21303

Appendix



Splitting

Merging

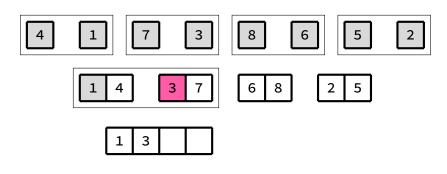
tion

Dronartia

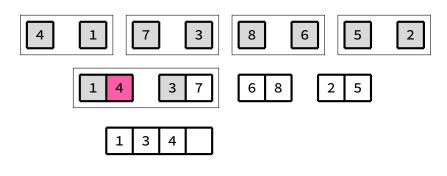
Sorting Lists

Bottom-Up

Appendix



Merging



Splitting

Merging Implemer

Analysi

Propertie:

Sorting Lists

Appendix

Merge Sort Demo

1 4 3

6 | 8

Splitting

Merging

Implemer tion

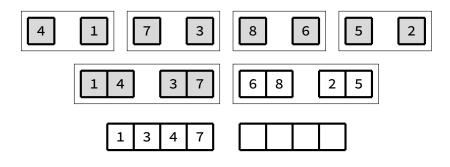
Analysis

Properties

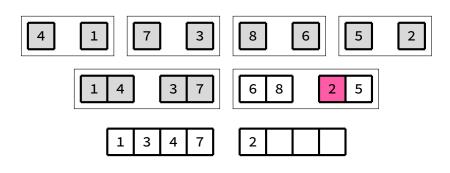
Sorting Lists

Annondin

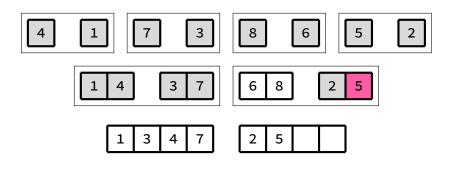
Appendix Marria Sort Dam



Merging



Merging



Splitting

Merging

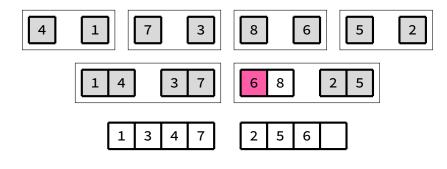
Impleme

Analysis

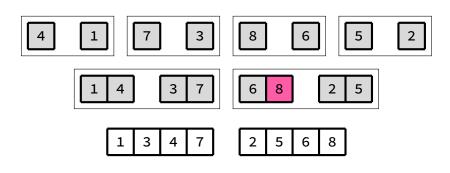
. .

0011115 2.50

Appendix



Merging



Method

Merging

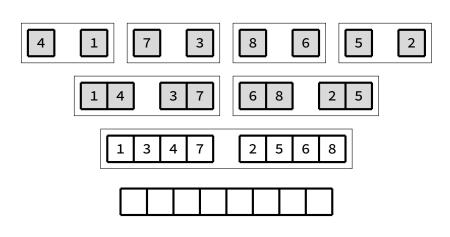
Implemen

Analysi

Properties

Sorting Lists

Appendix



Colitting

Merging

Implemen

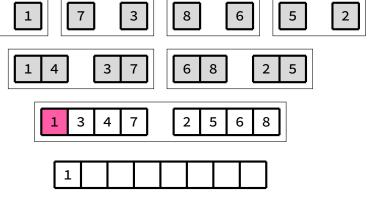
Analysi

Propertie

Sorting Lists

Dotton IIn

Appendix



Colitting

Merging

Impleme

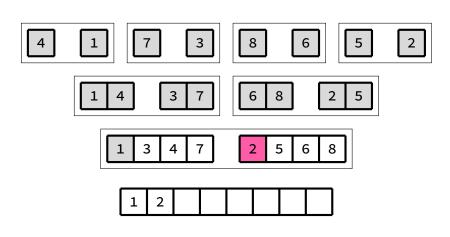
Analysis

rioperties

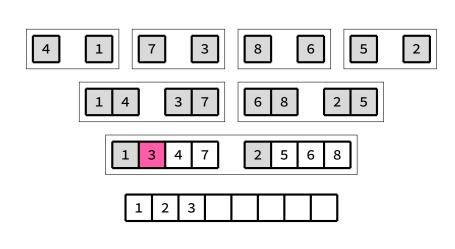
Sorting Lists

Annondiv

Appendix



Merging



Colitting

Merging

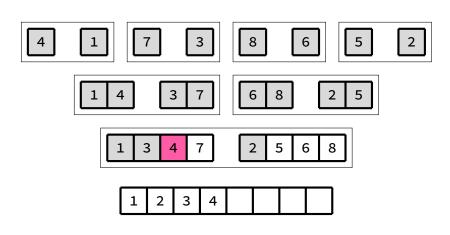
Implemention

Analysis

Properties

Sorting Lists

Appendix



Method

Merging

Implemen

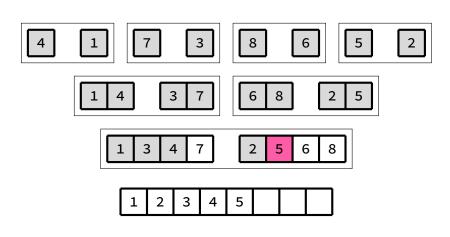
Analysis

Properties

Sorting Lists

A no no no disc

Appendix



Method

Merging

Implome

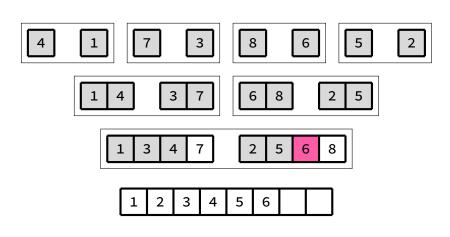
Analysi

Properties

Sorting Lists

BULLUIII-UP

Appendix



Method

Merging

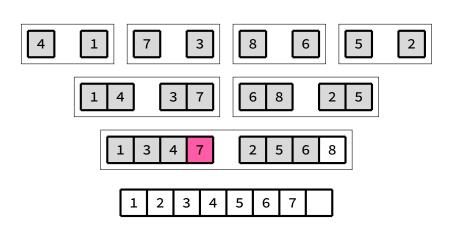
Implemer

Analysi

Properties

Sorting Lists

Appendix



### Merge Sort

Splitting

Merging

Impleme

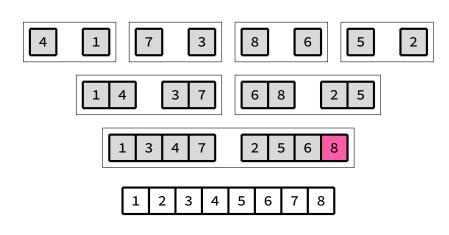
Analysi

rioperties

Sorting Lists

. . . . . . .

Appendix



### Merge Sort

Splitting

Merging

Implemen

Analysi

Properties

Sorting Lists

DOLLOIII-OP

Appendix

