Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Slides by **Sean Szumlanski**

for **COP 3502**, Computer Science I

Summer 2018

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

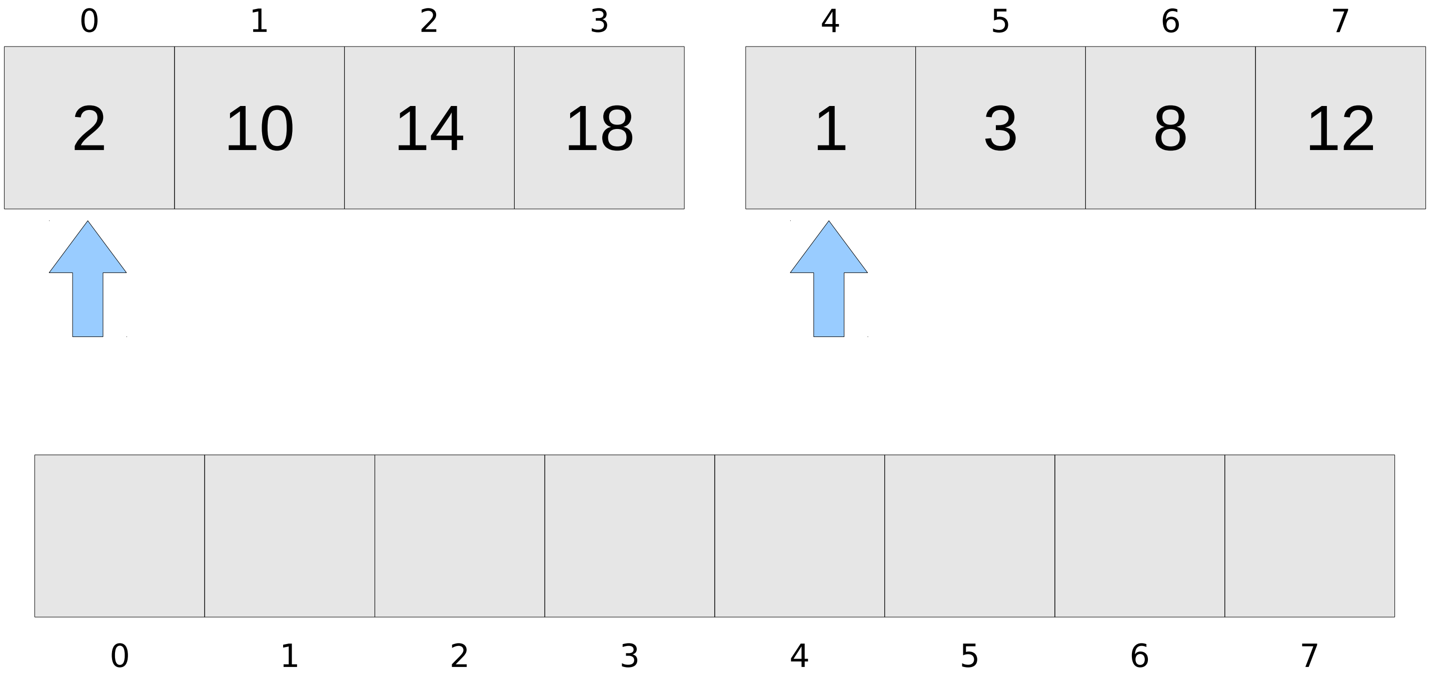
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



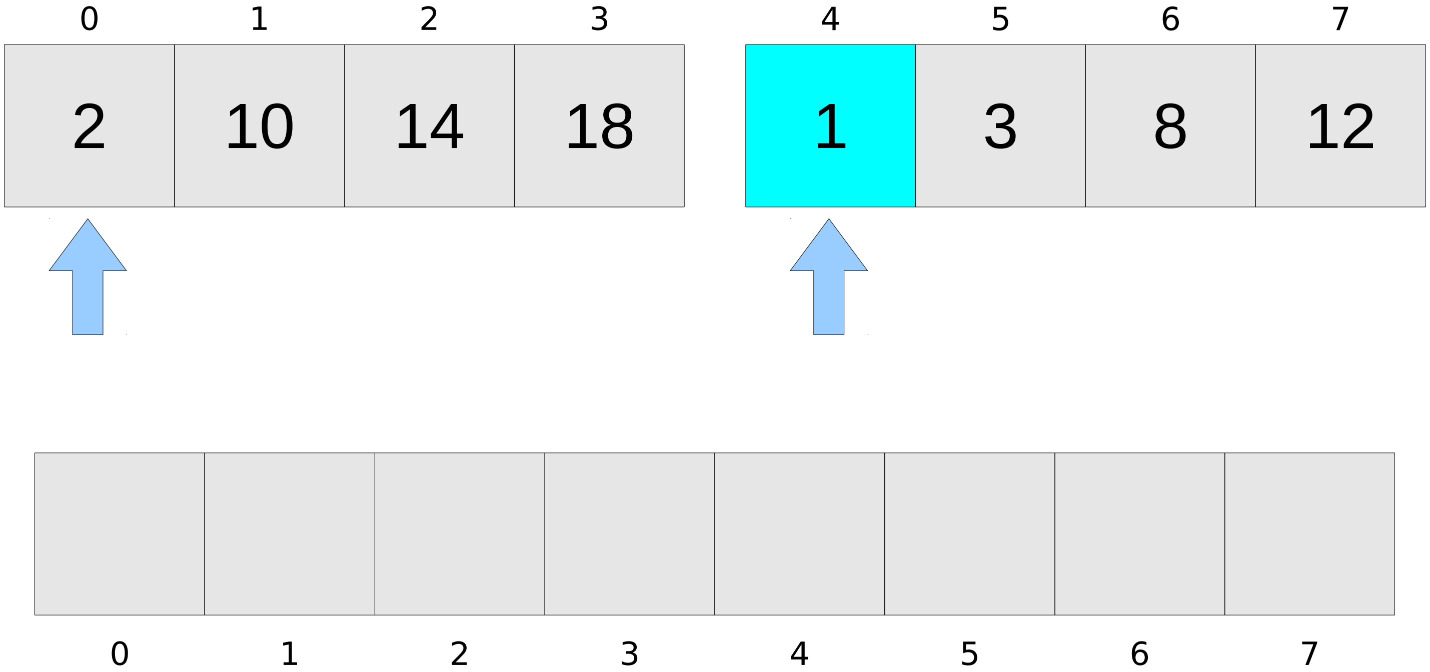
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

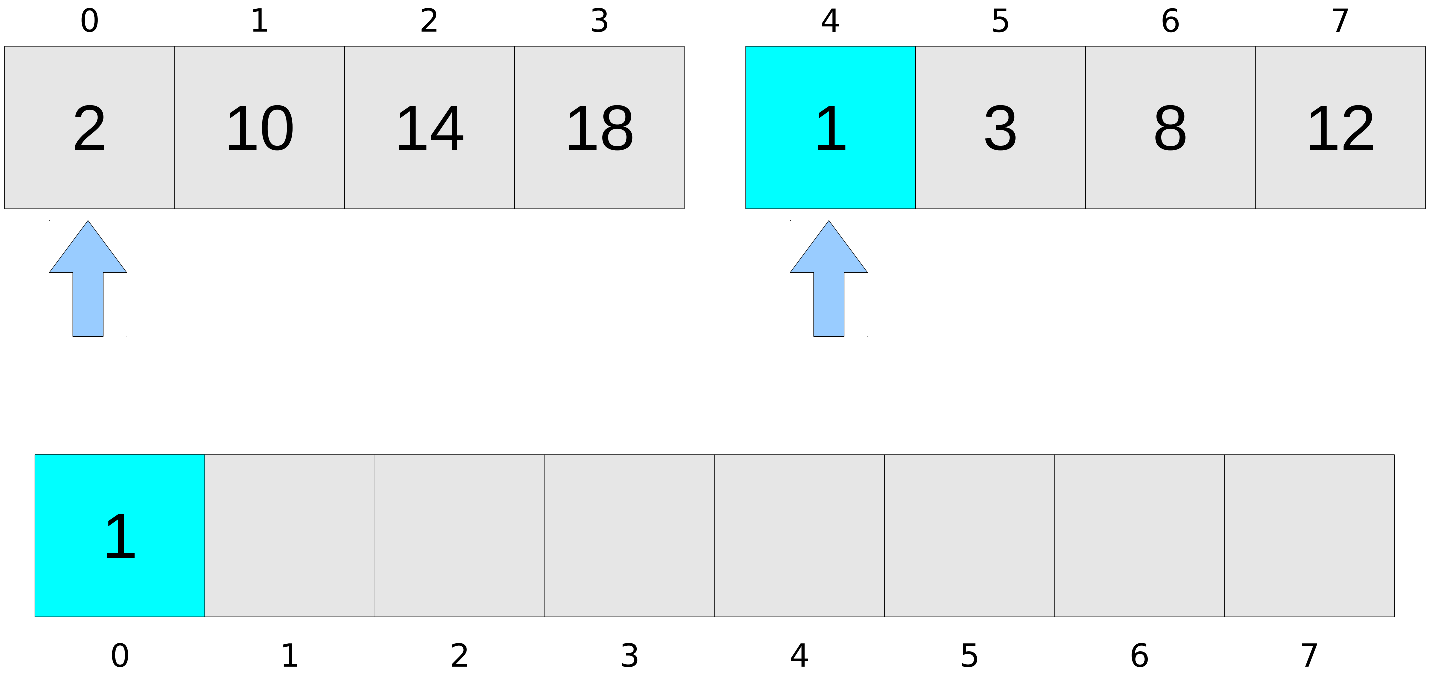


0	1	2	3	4	5	6	7
2	10	14	18	1	3	8	12

0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

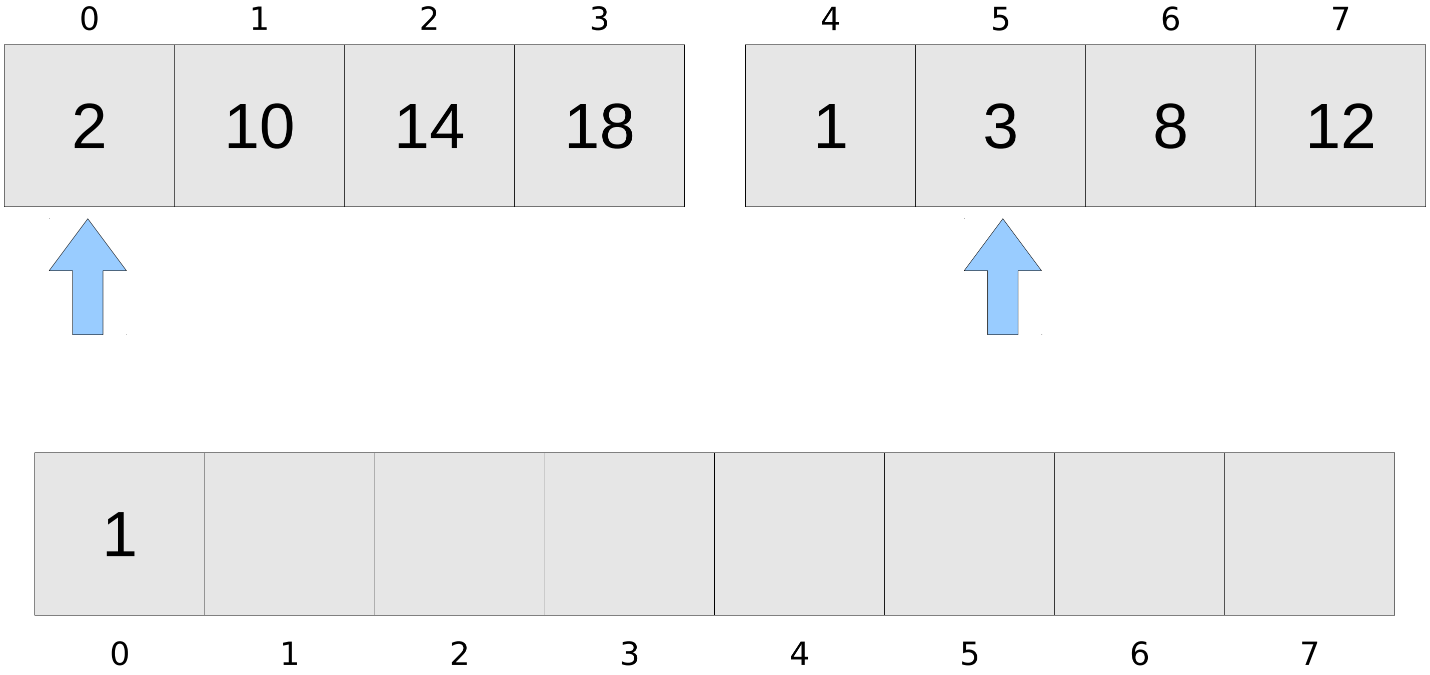


0	1	2	3	4	5	6	7
2	10	14	18	1	3	8	12

1							
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



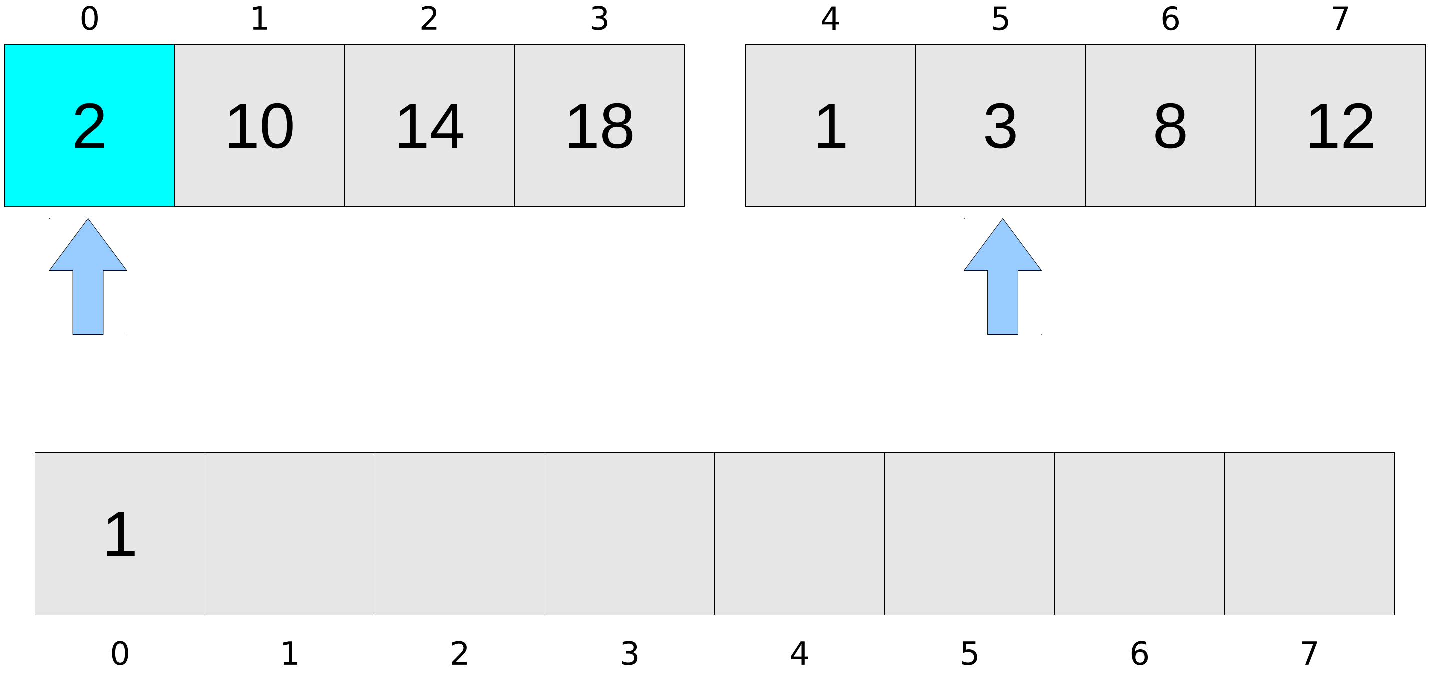
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1							
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



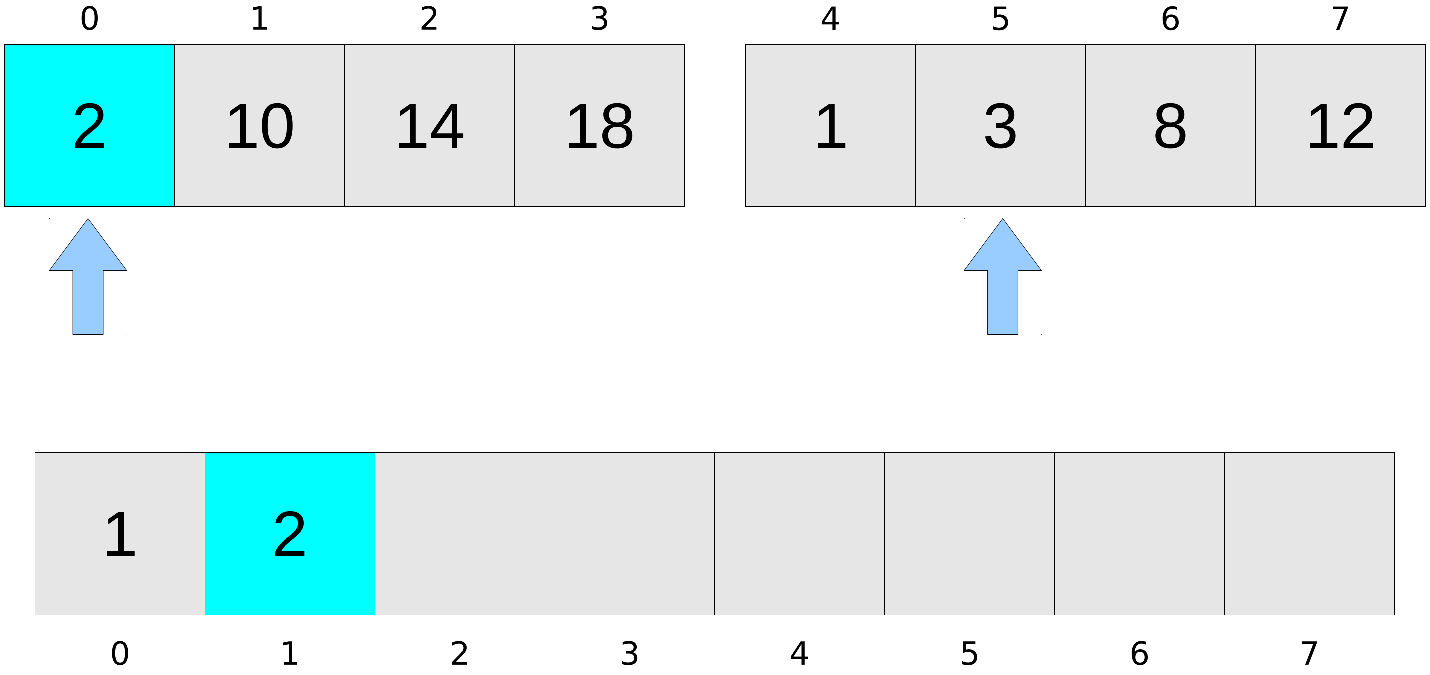
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1							
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



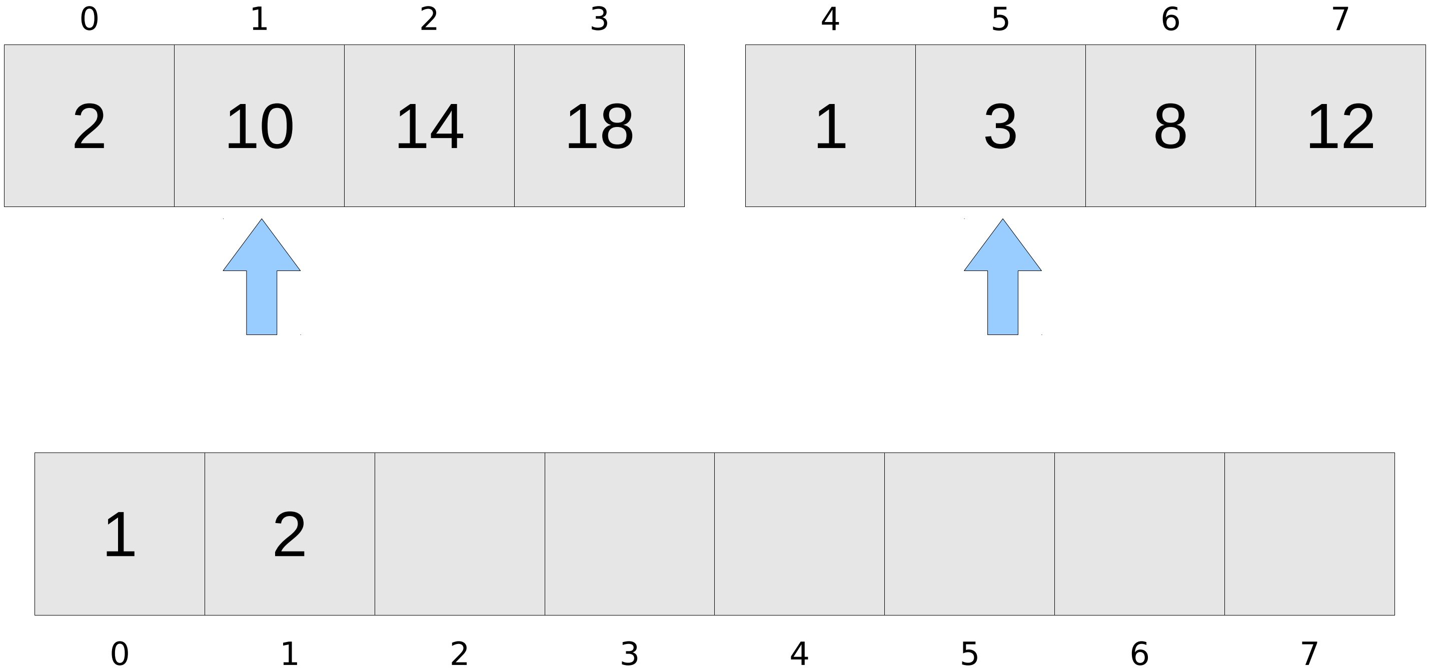
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2						
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



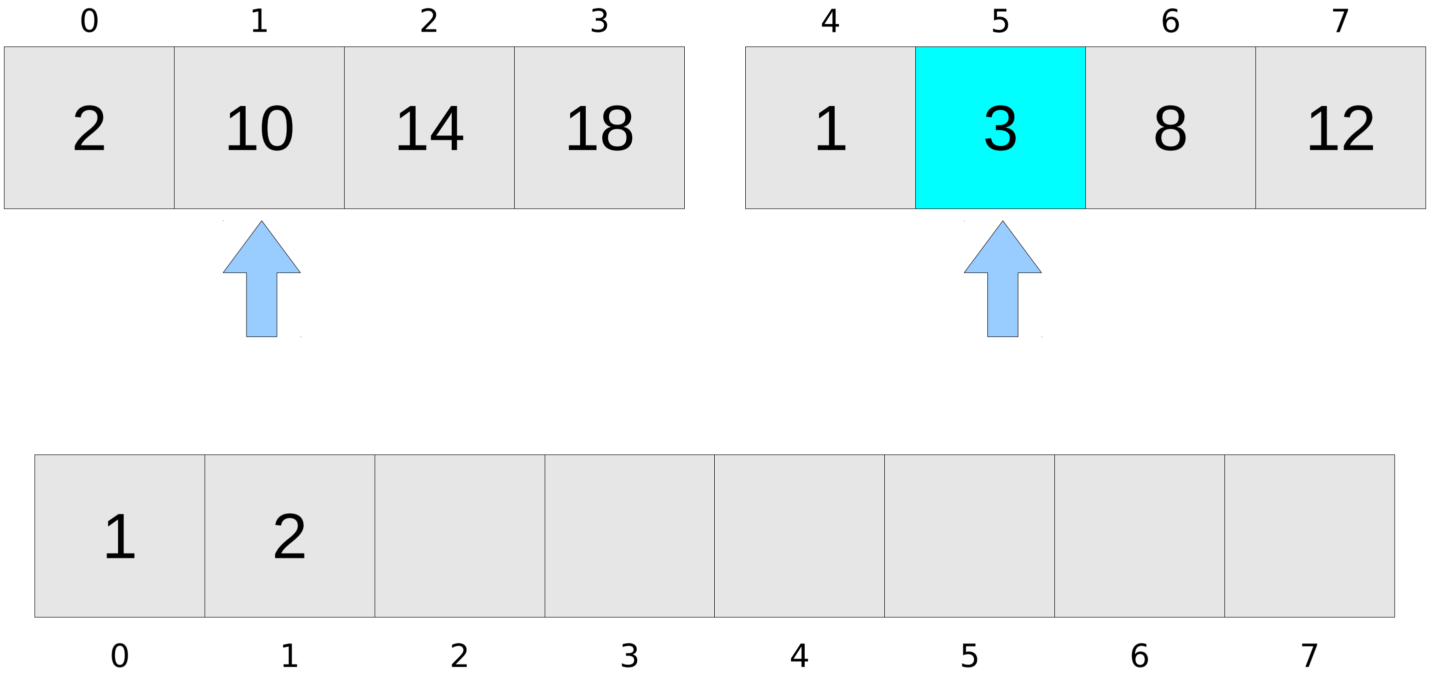
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2						
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



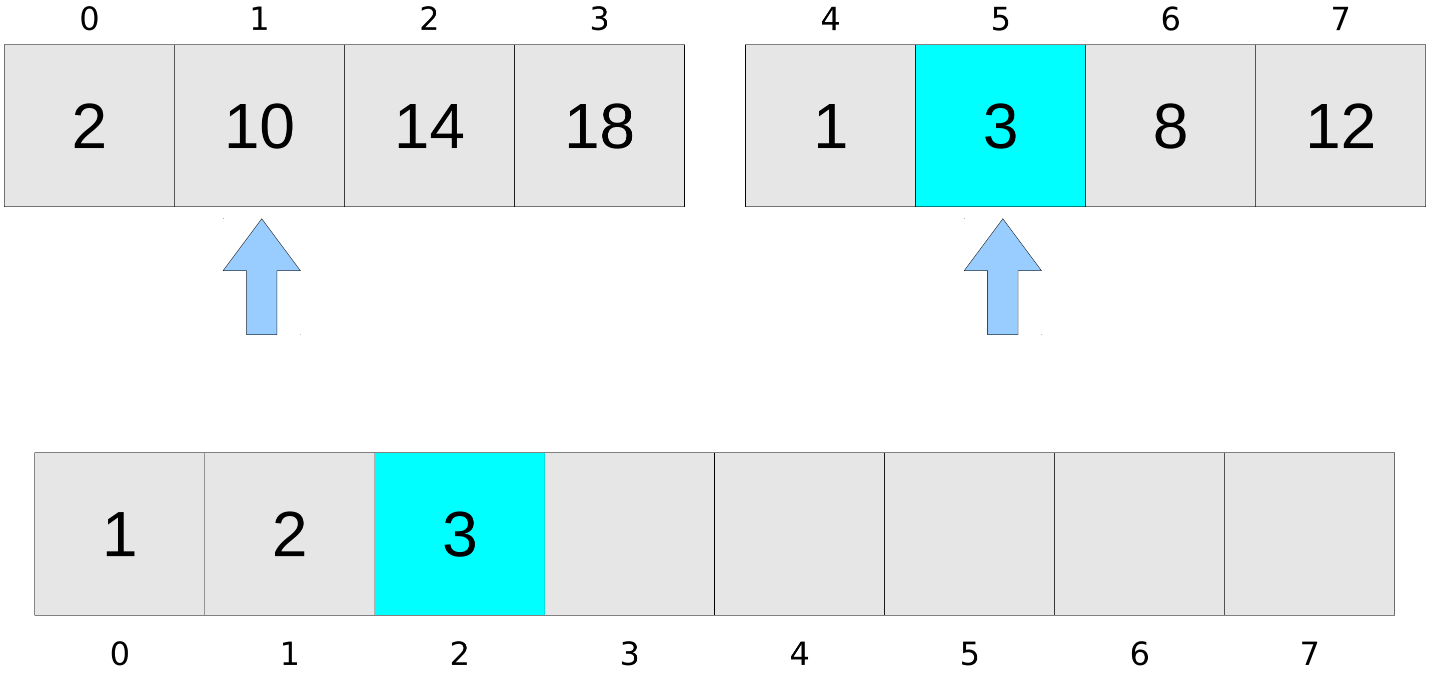
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2						
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



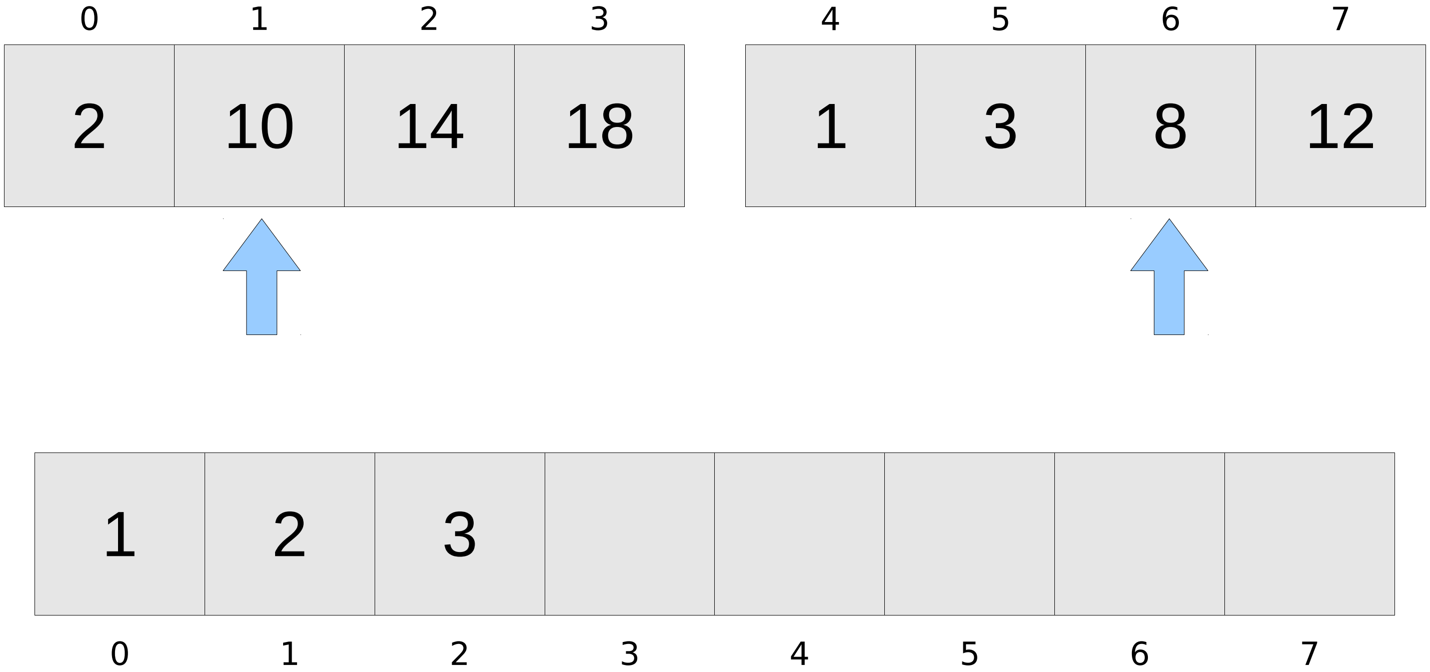
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3					
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



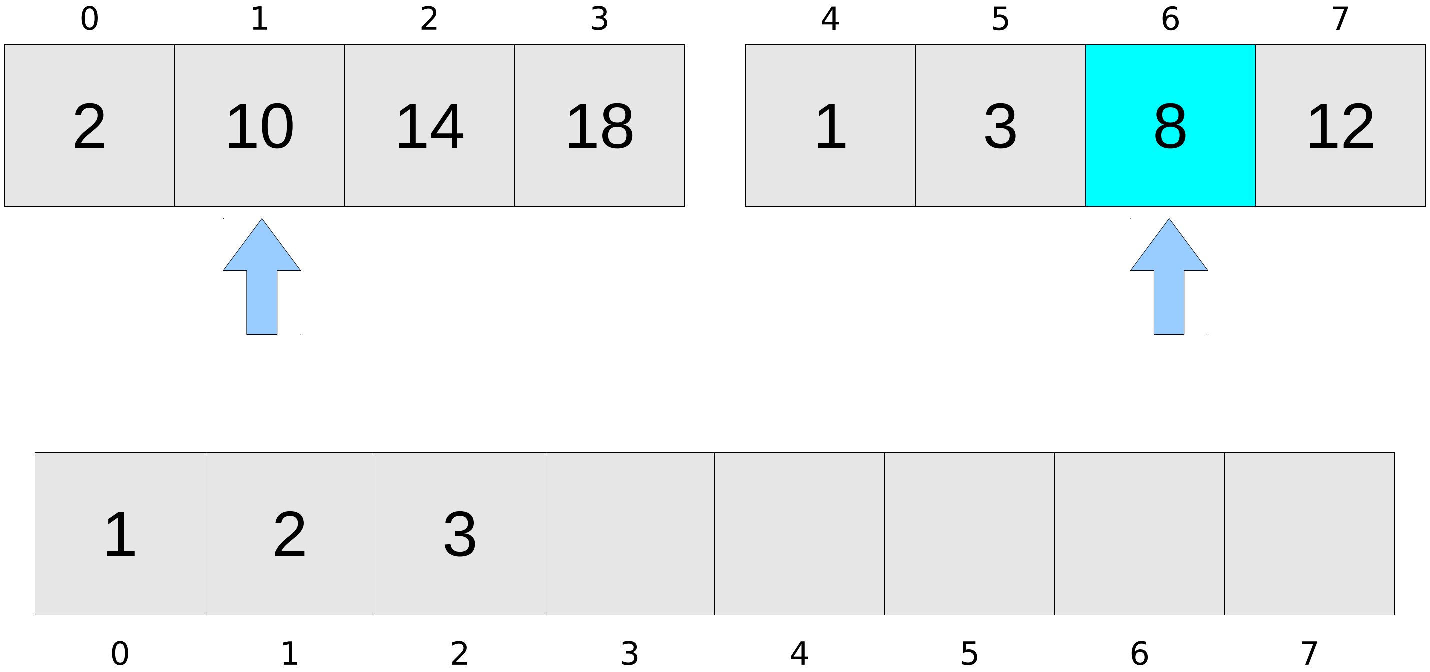
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3					
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



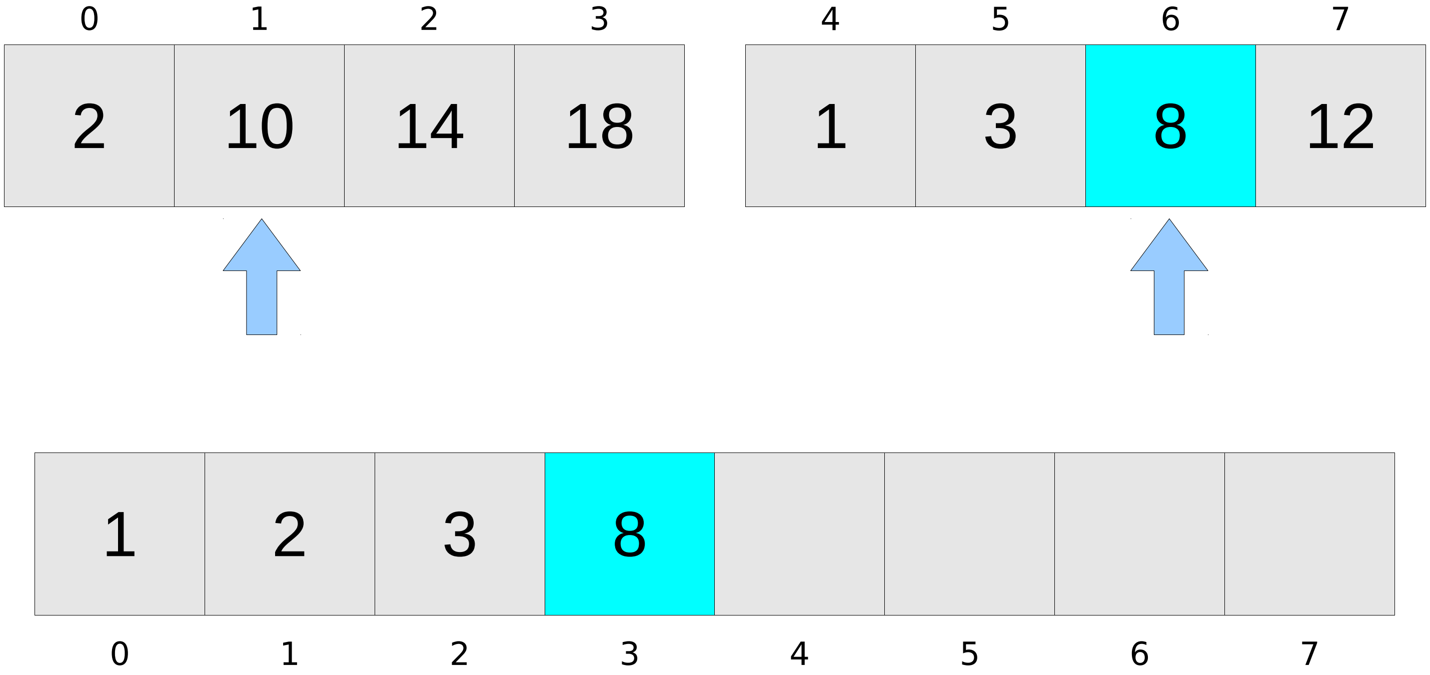
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1	2	3	4	5	6	7
1	2	3					

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



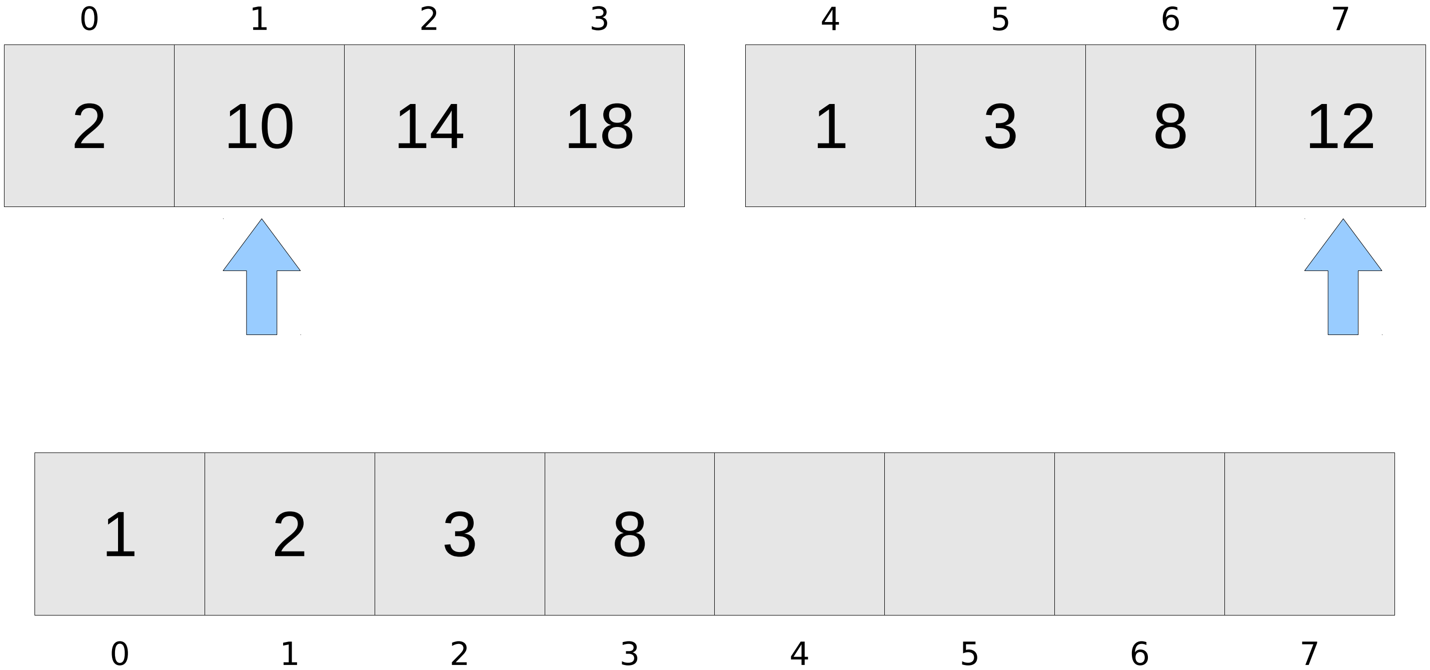
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1	2	3	4	5	6	7
1	2	3	8				

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



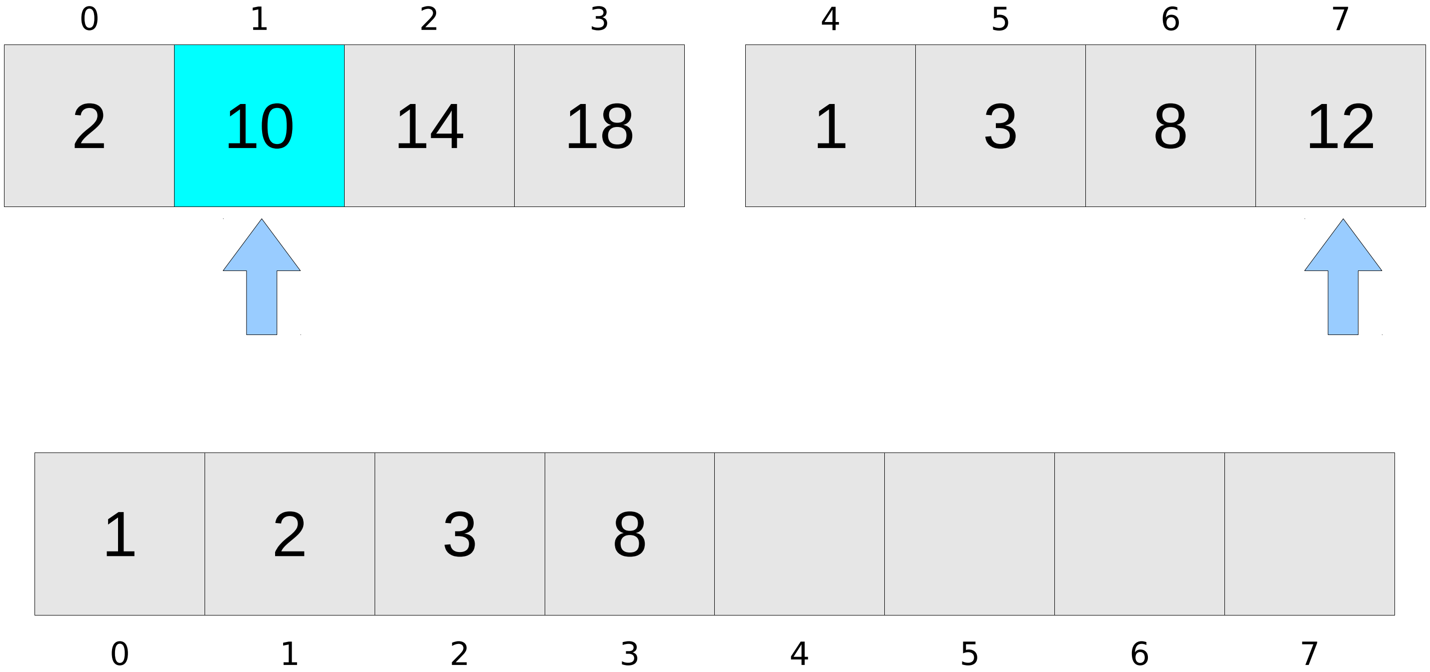
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8				
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



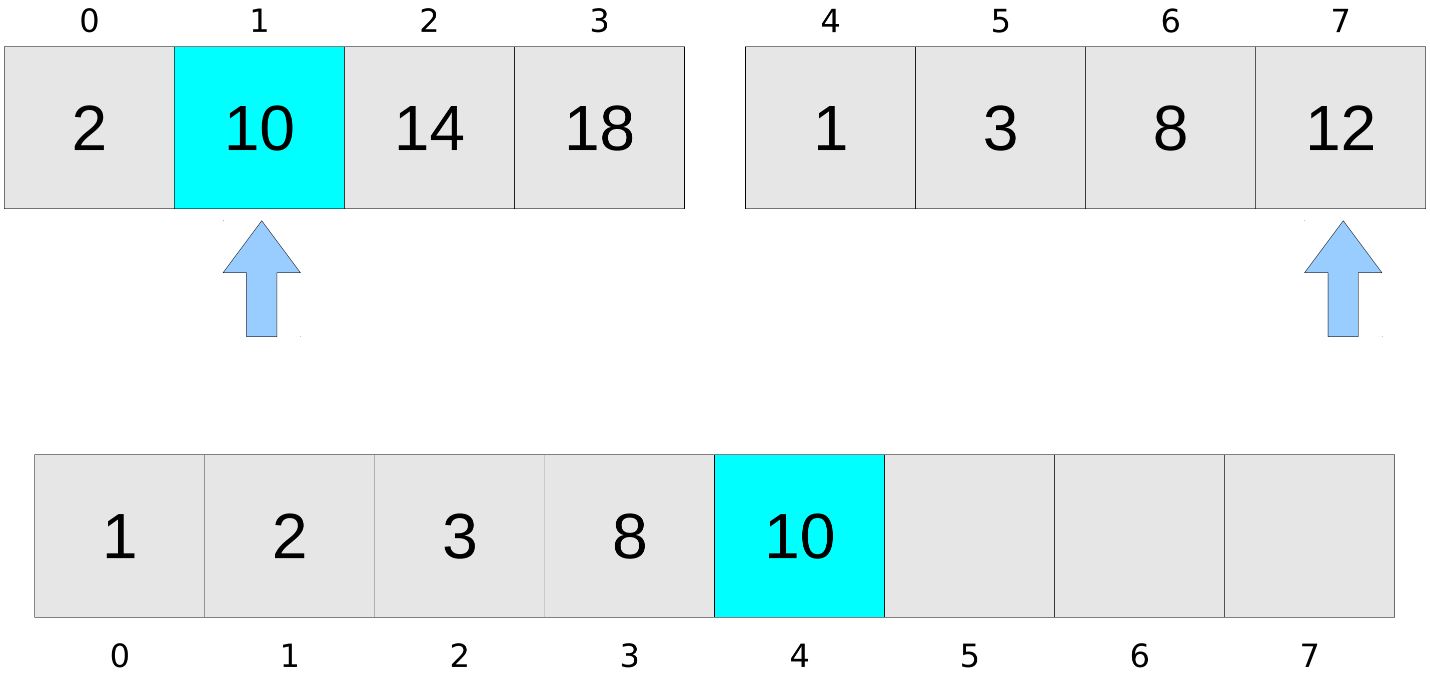
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8				
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



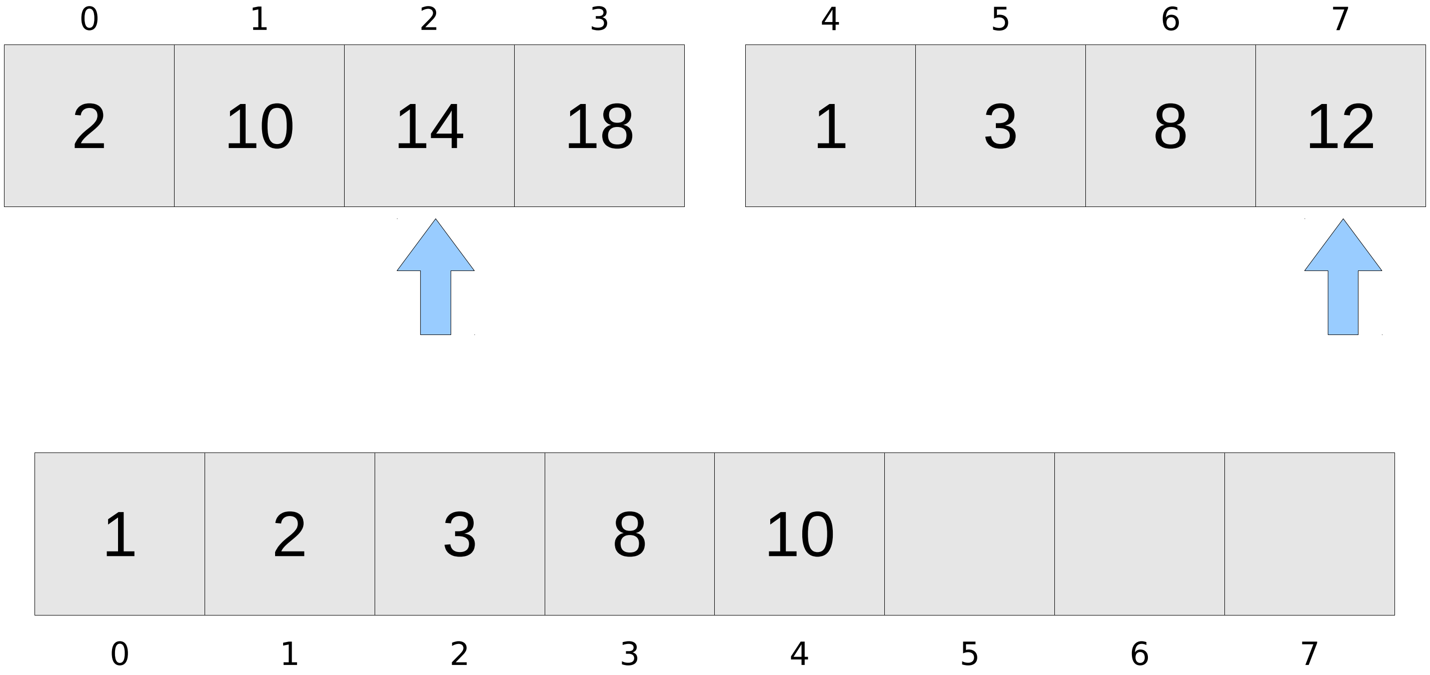
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10			
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



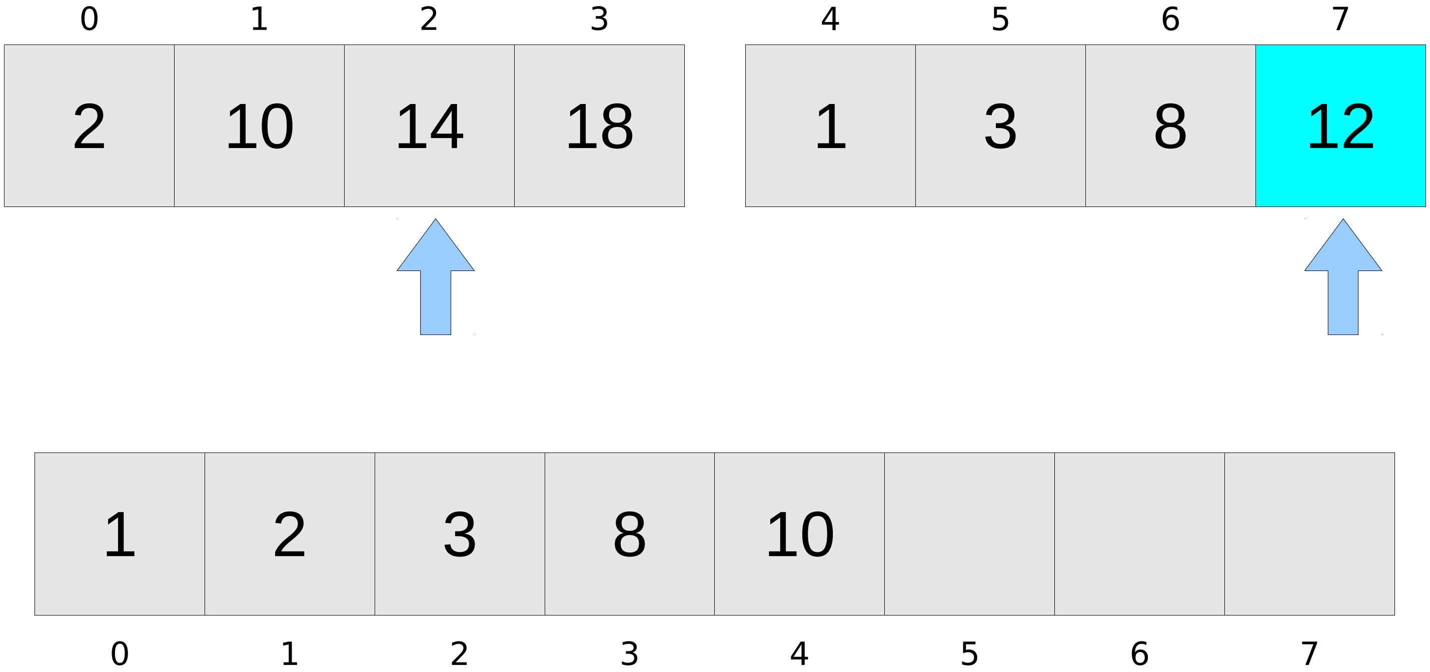
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1	2	3	4	5	6	7
1	2	3	8	10			

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



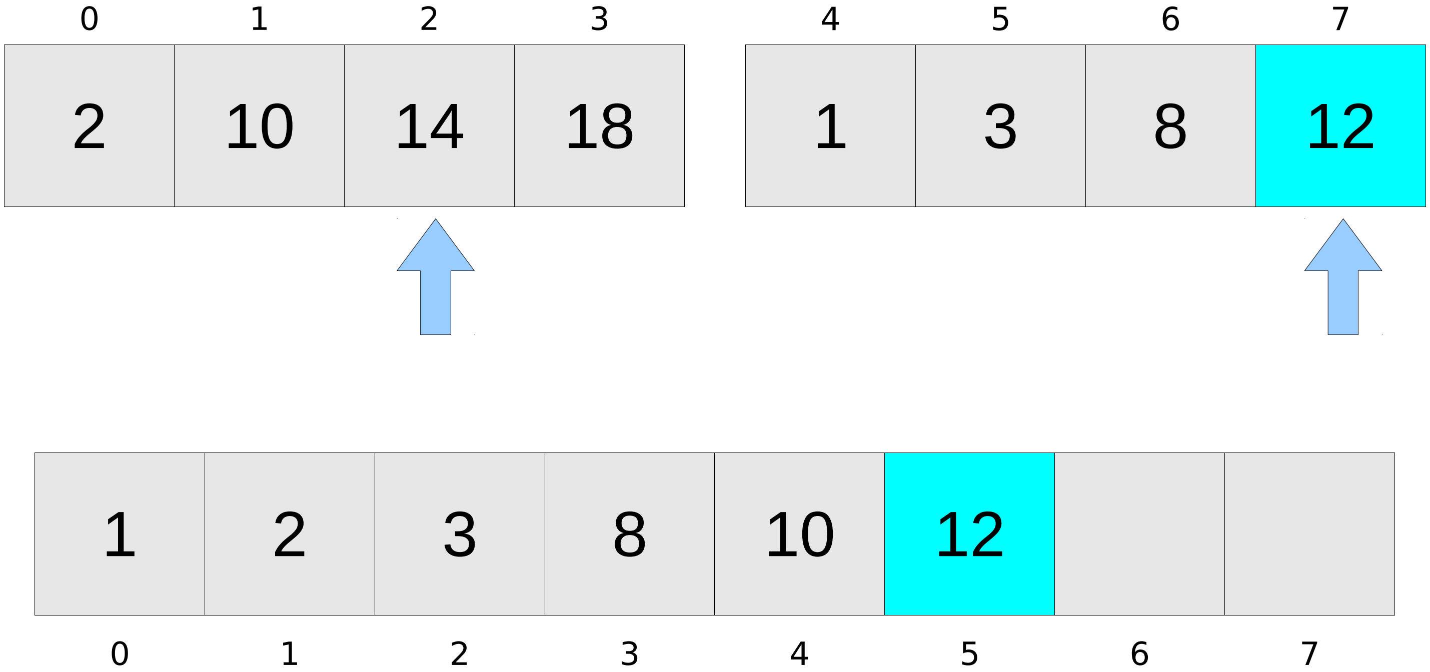
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10			
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

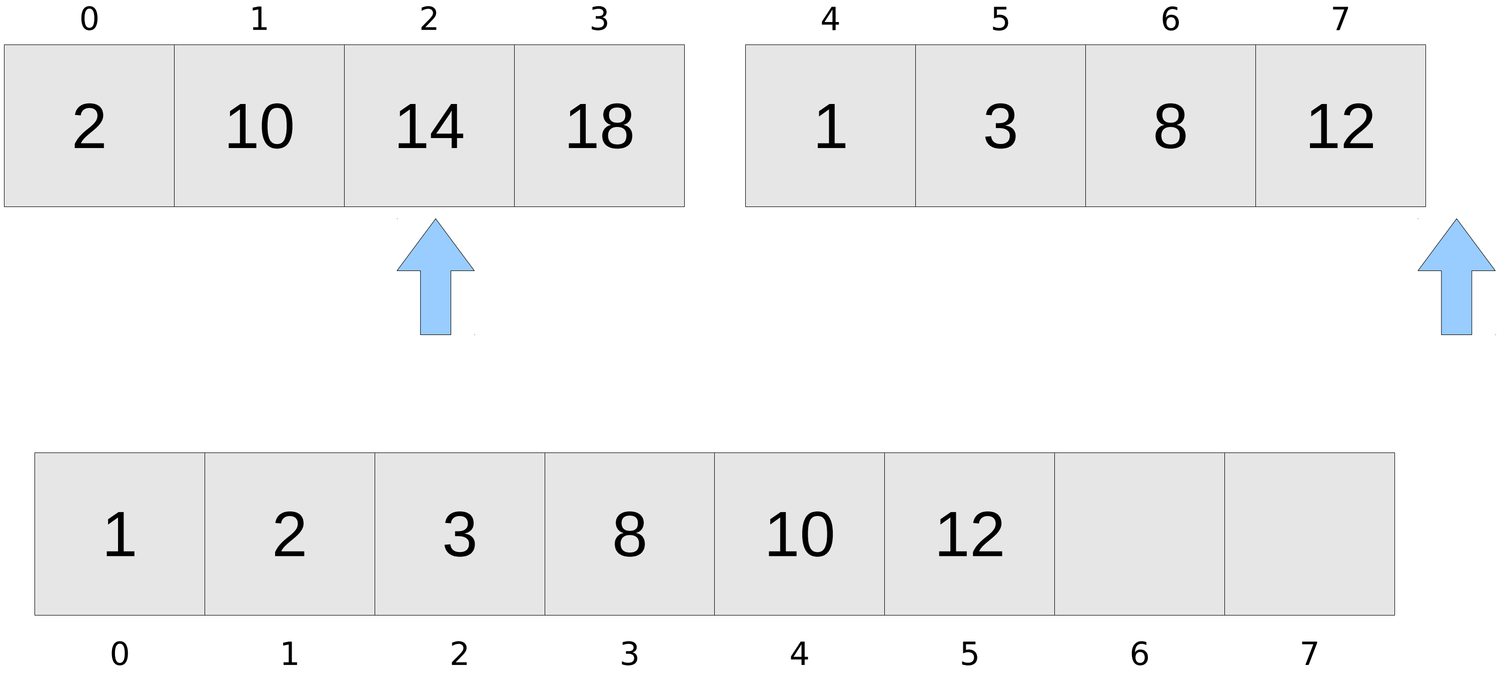


0	1	2	3	4	5	6	7
2	10	14	18	1	3	8	12

1	2	3	8	10	12		
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

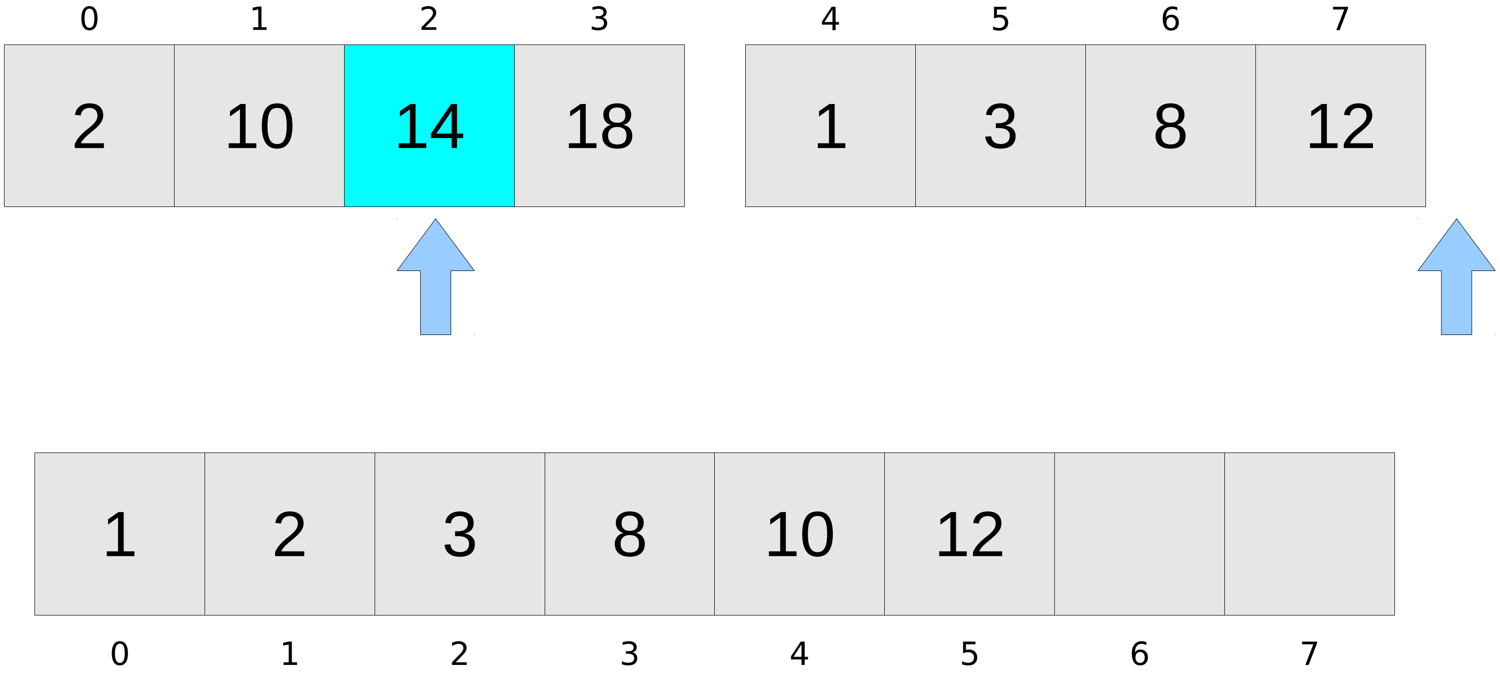


0	1	2	3	4	5	6	7
2	10	14	18	1	3	8	12

1	2	3	8	10	12		
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



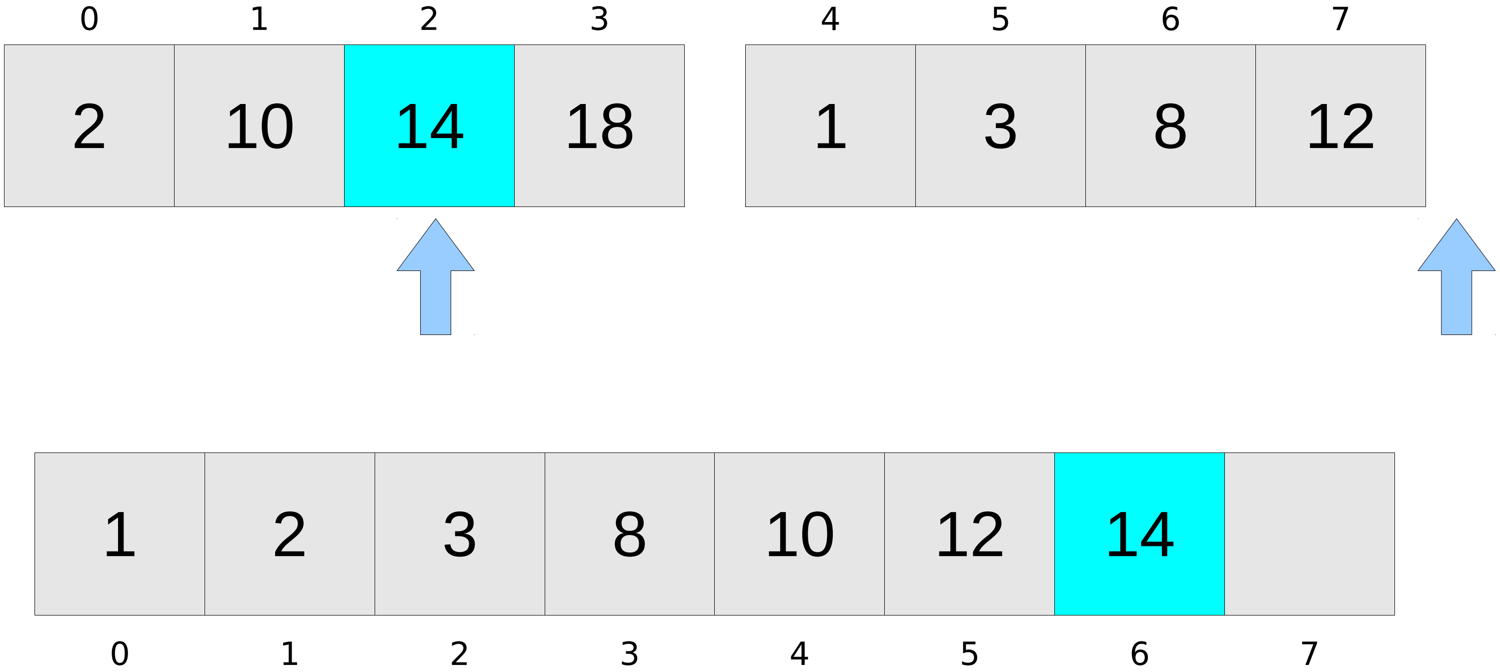
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12		
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



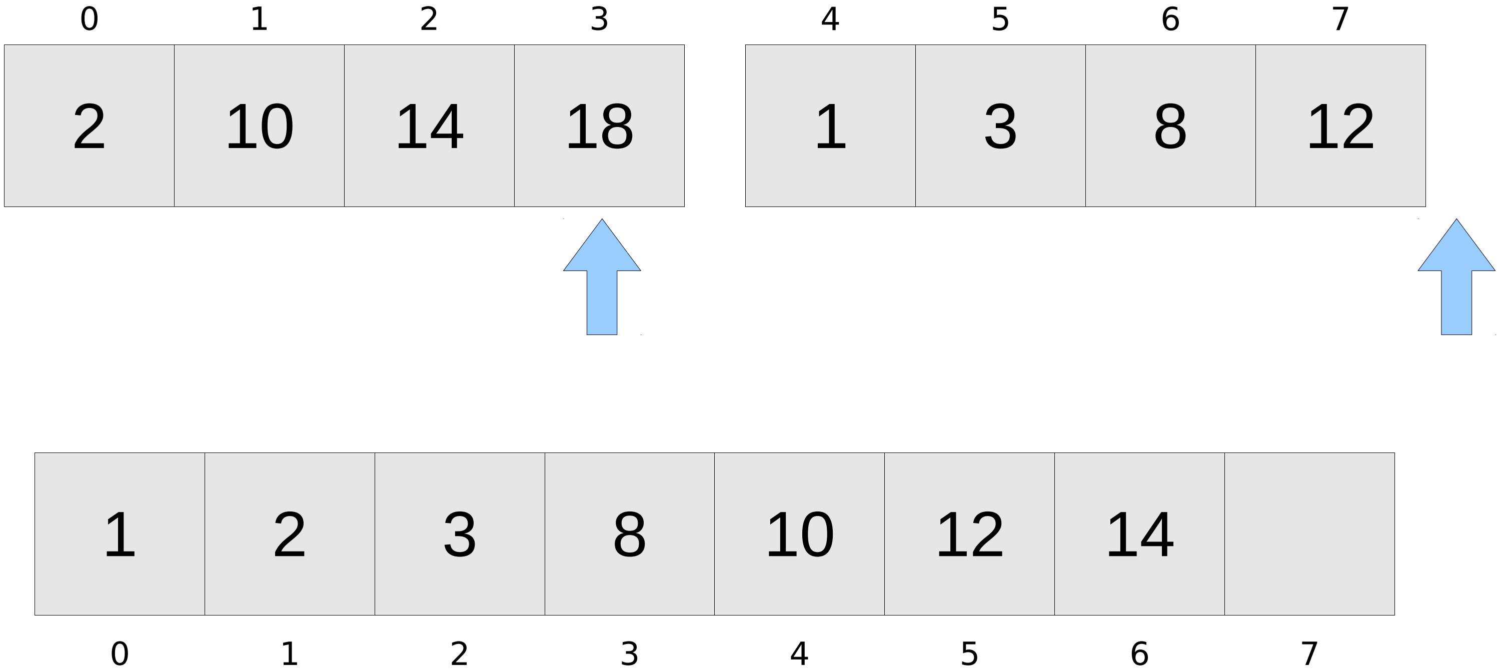
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12	14	
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



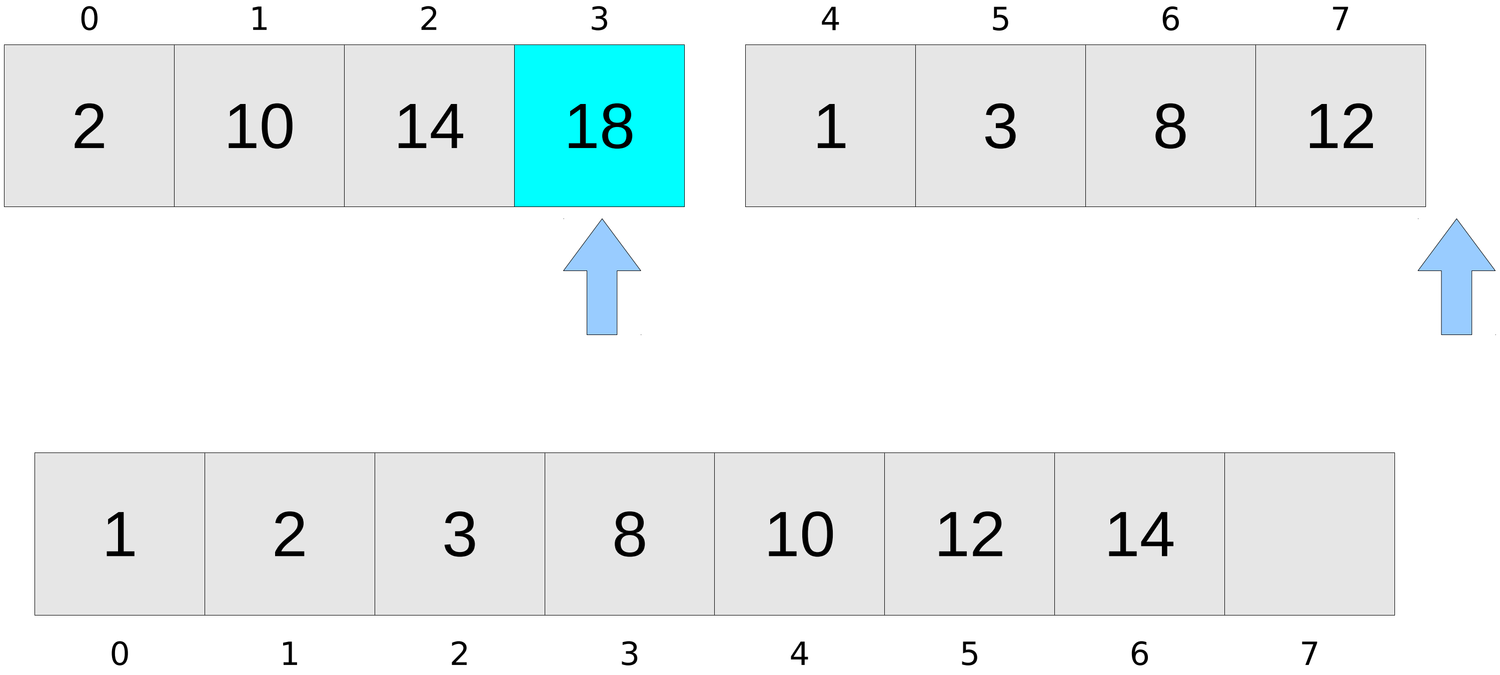
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12	14	
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



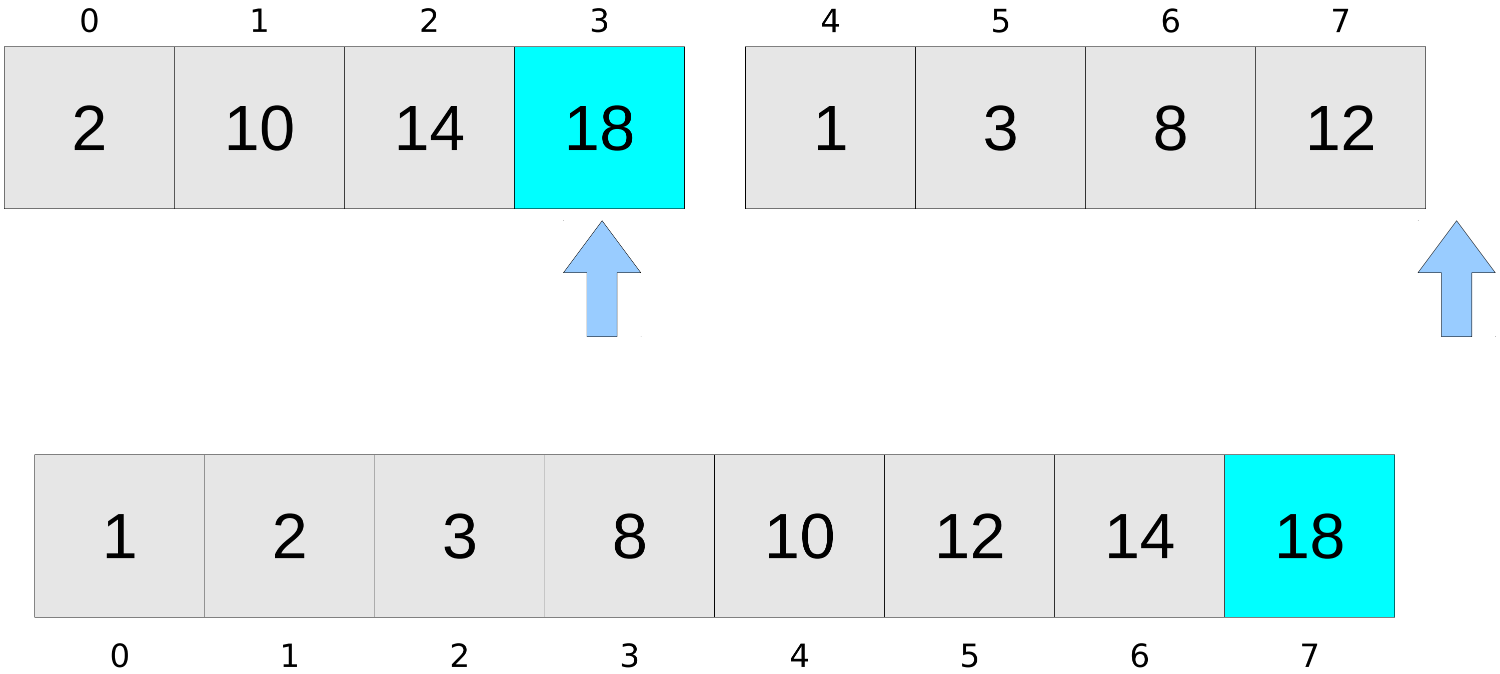
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12	14	
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



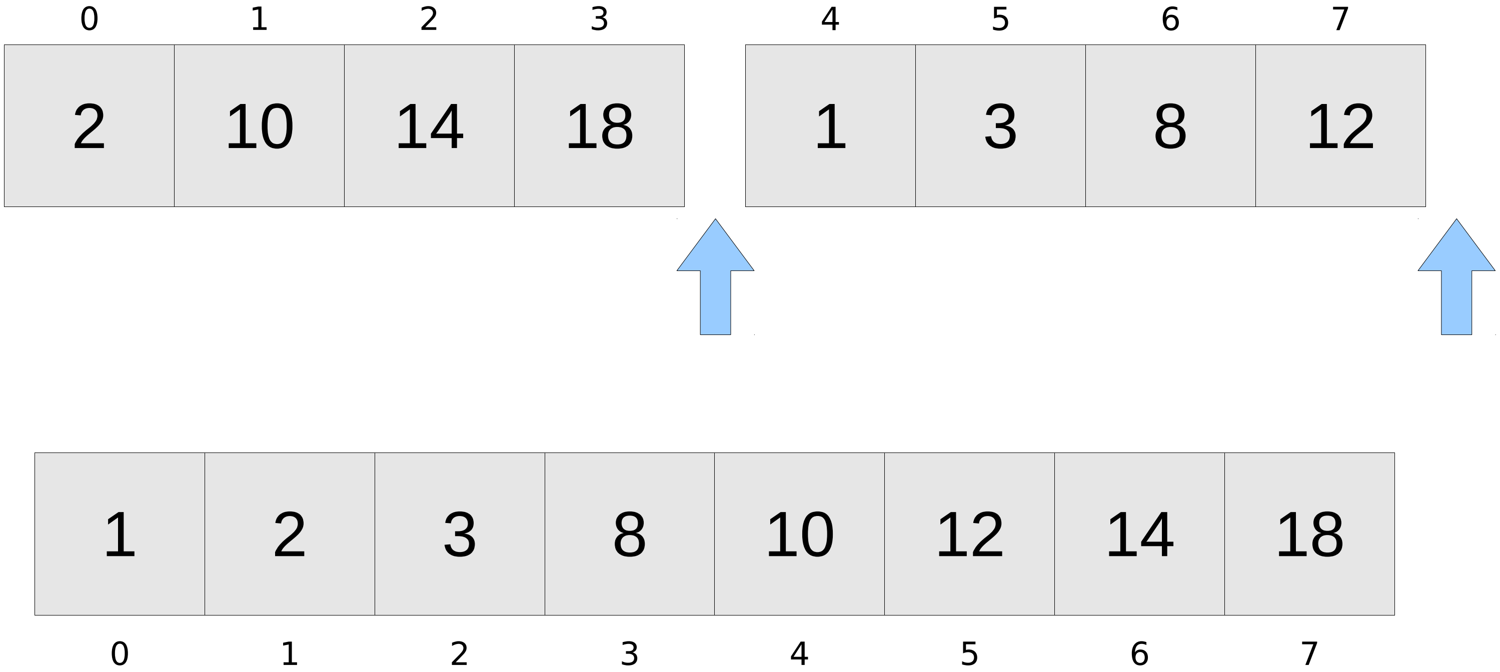
0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12	14	18
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?



0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

1	2	3	8	10	12	14	18
0	1	2	3	4	5	6	7

But First of All...

Do you agree we can merge two **sorted** arrays into **one** sorted array in **linear** time?

0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

TADA

1	2	3	8	10	12	14	18
0	1	2	3	4	5	6	7

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Merge Sort

(break down the array, then merge the pieces back together)

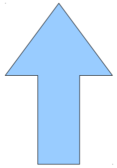
0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Recall: mid = lo + (hi – lo) / 2;

Merge Sort

(break down the array, then merge the pieces back together)

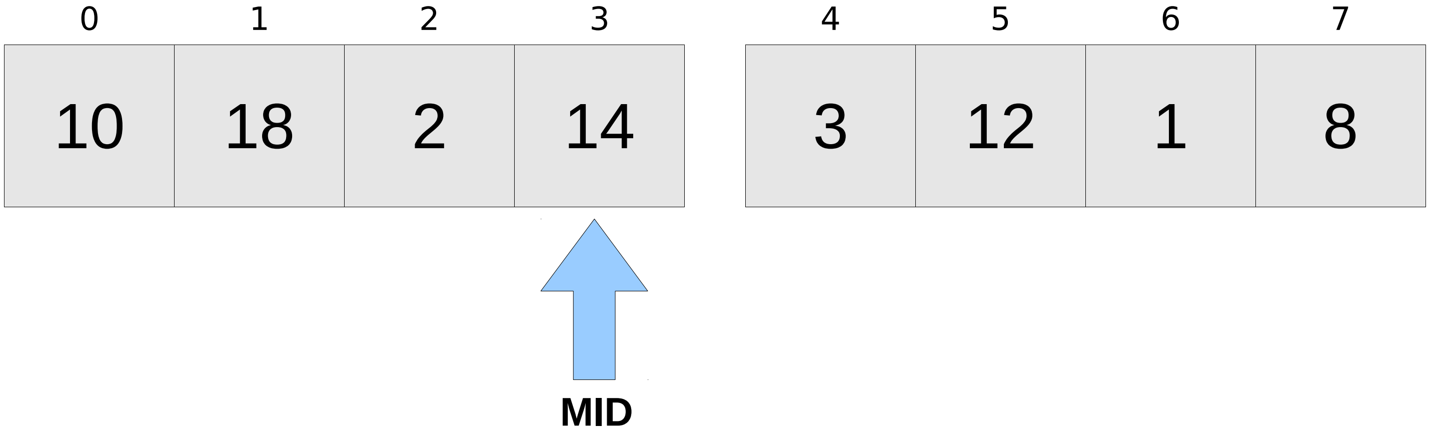
0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8



Recall: mid = lo + (hi – lo) / 2;

Merge Sort

(break down the array, then merge the pieces back together)

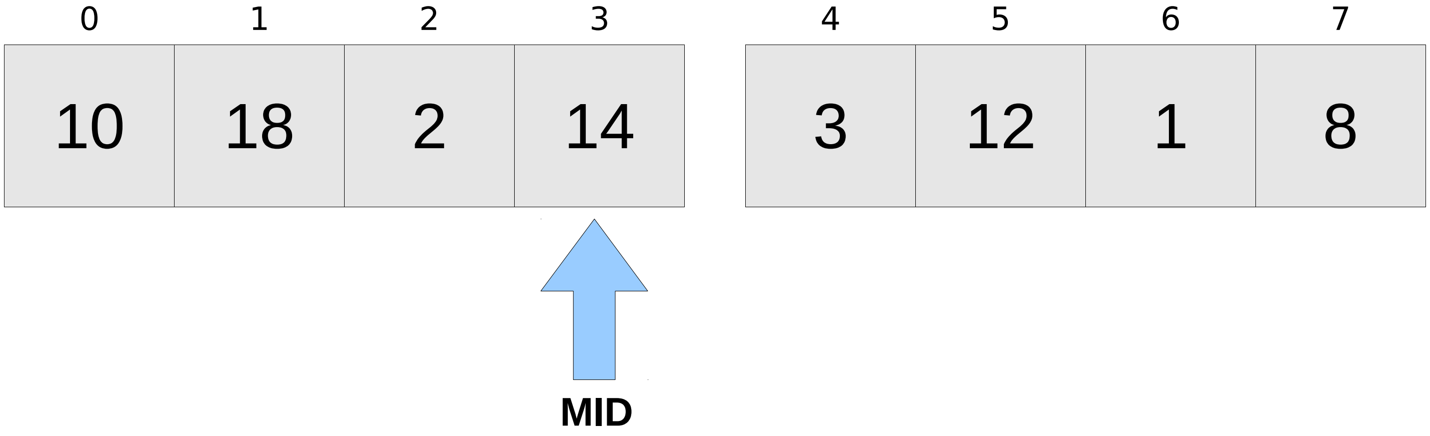


0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Recall: mid = lo + (hi – lo) / 2;

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Recall: mid = lo + (hi – lo) / 2;

MergeSort(array, lo, hi) → **MergeSort**(array, lo, mid)
MergeSort(array, mid + 1, hi)

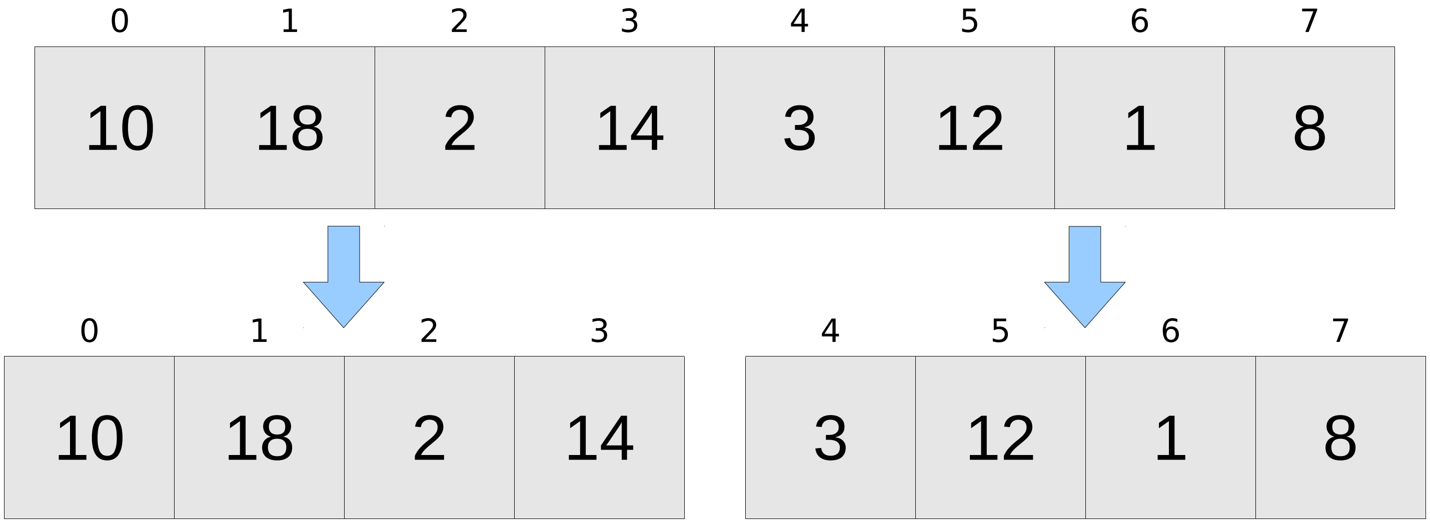
Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

Merge Sort

(break down the array, then merge the pieces back together)



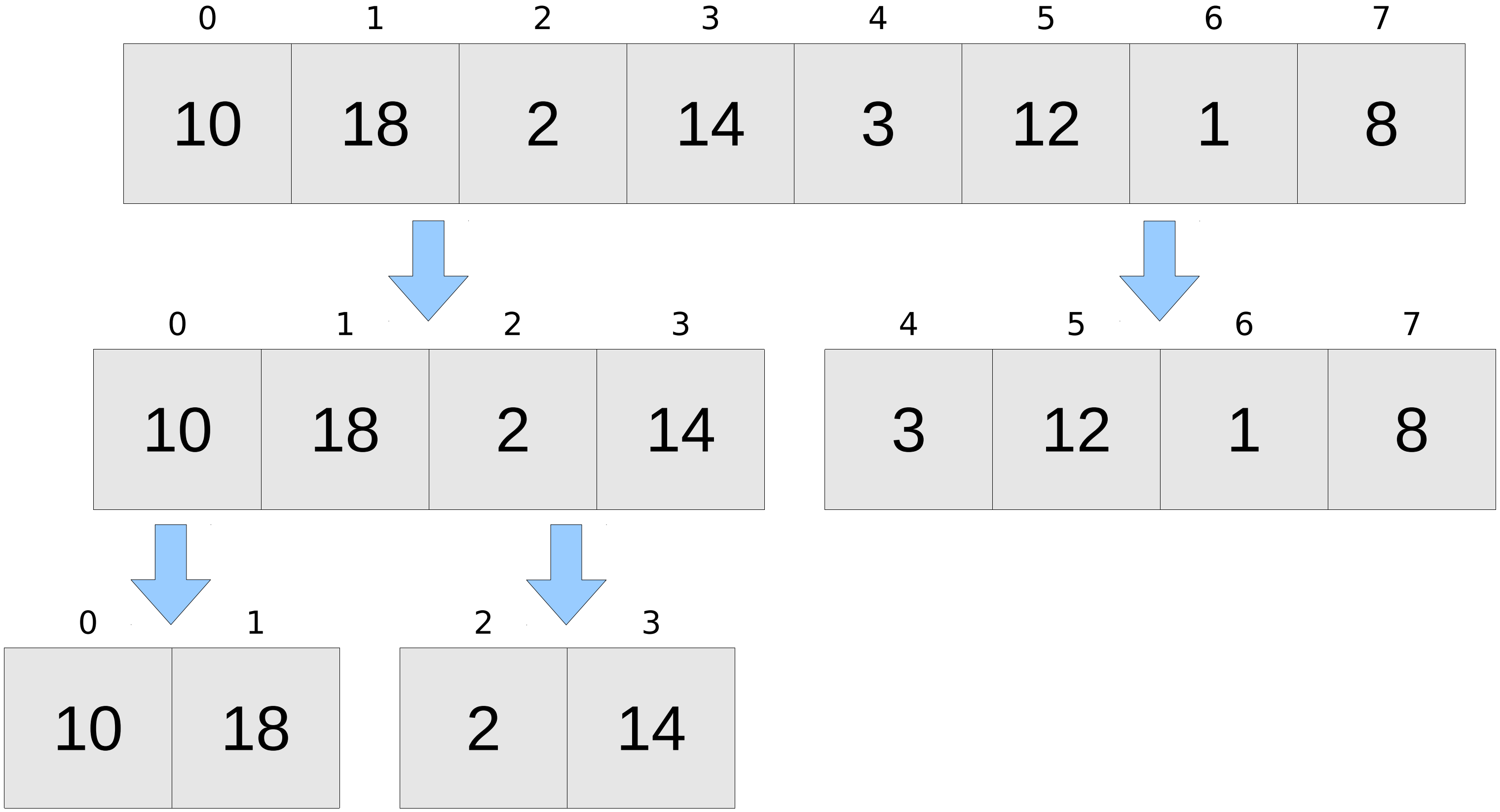
0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

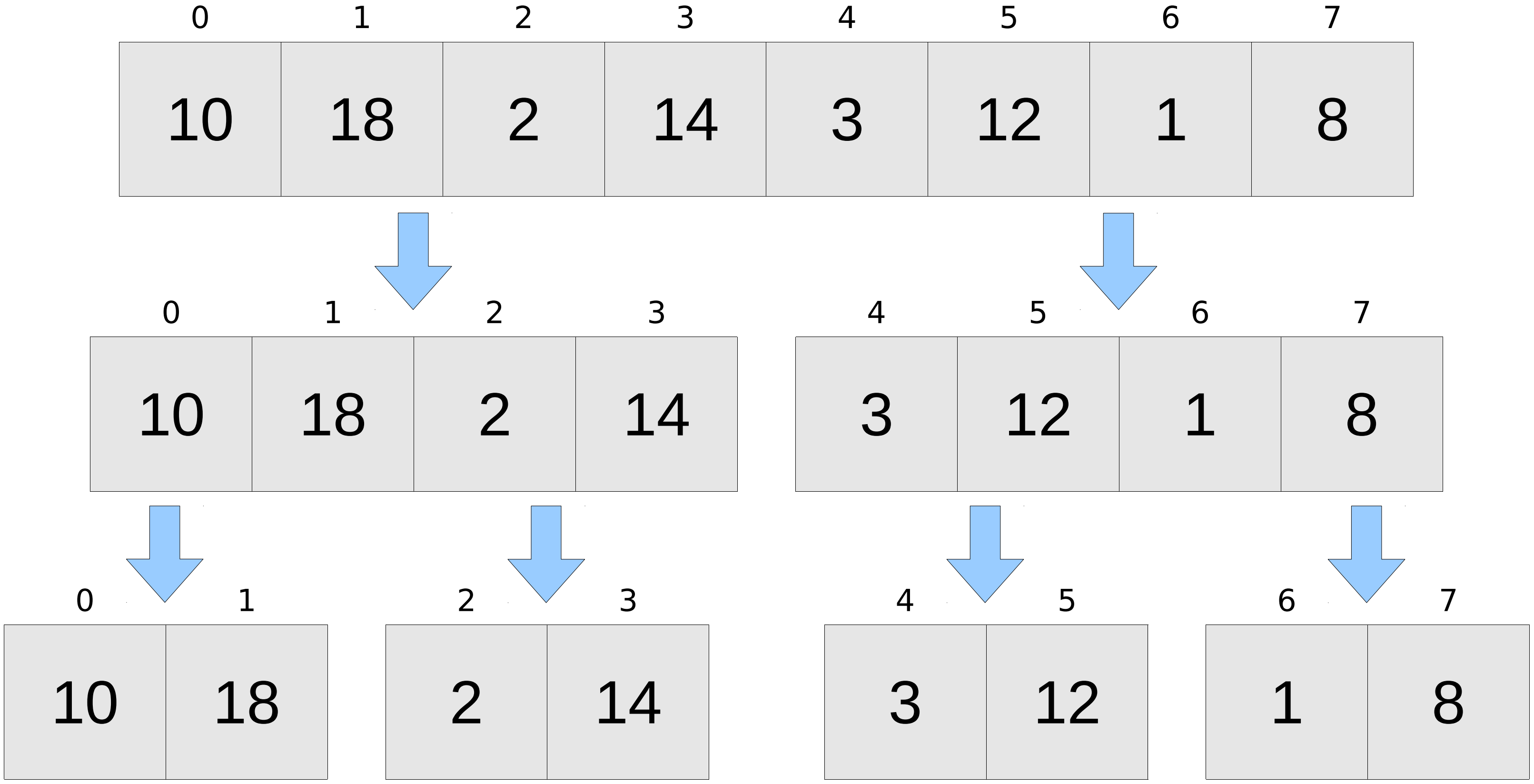
4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

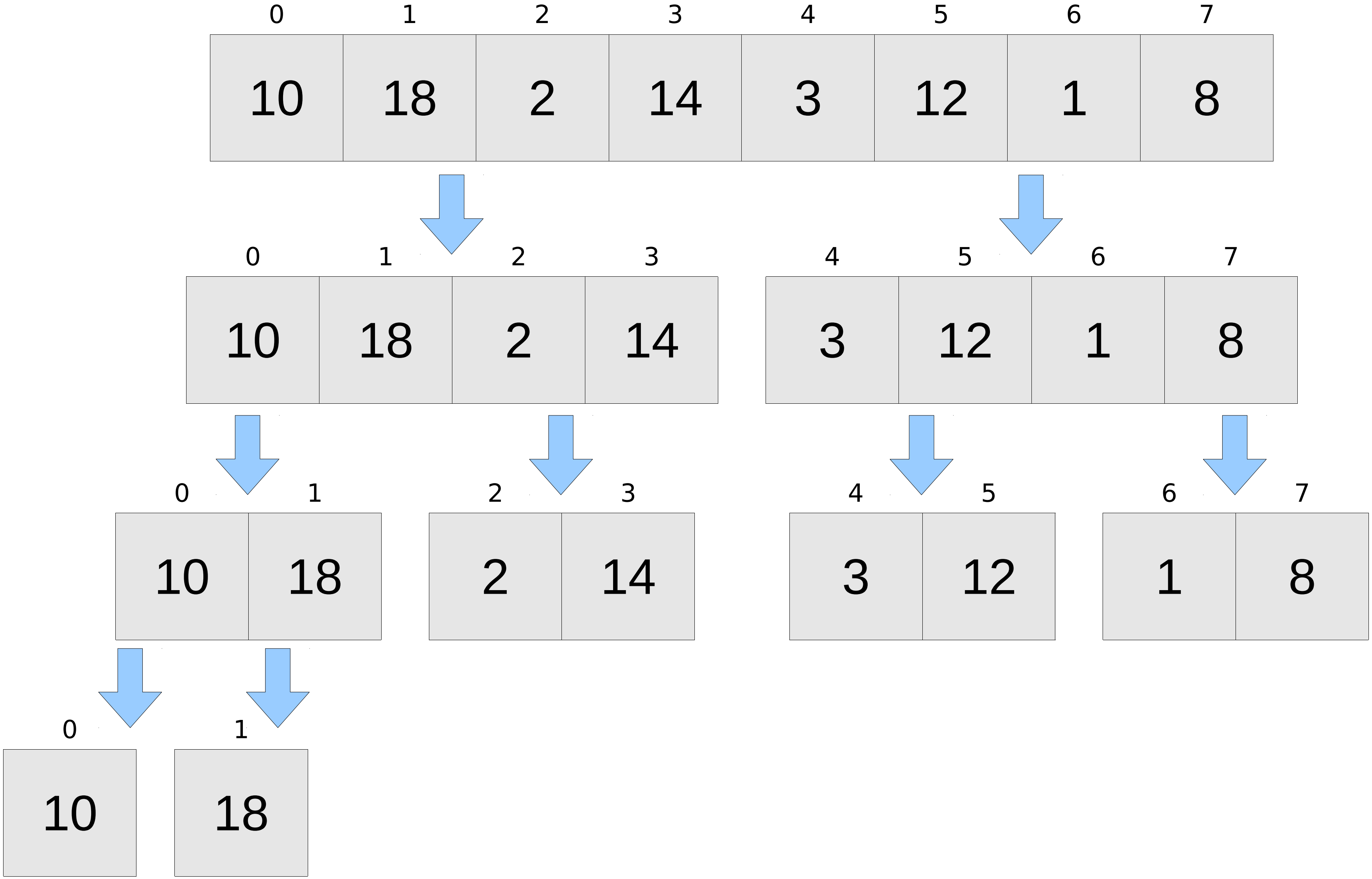
2	3
2	14

4	5
3	12

6	7
1	8

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

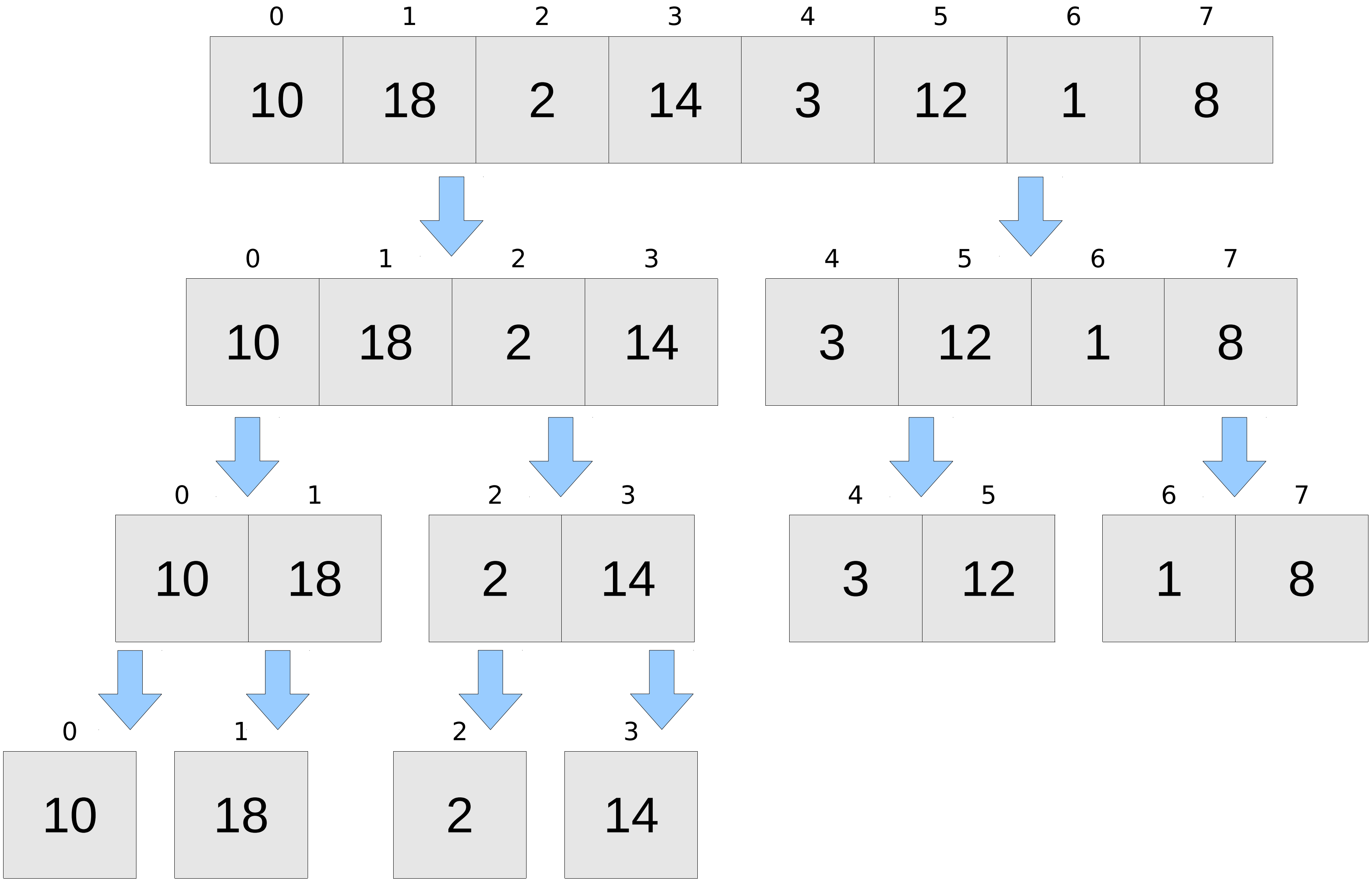
6	7
1	8

0
10

1
18

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

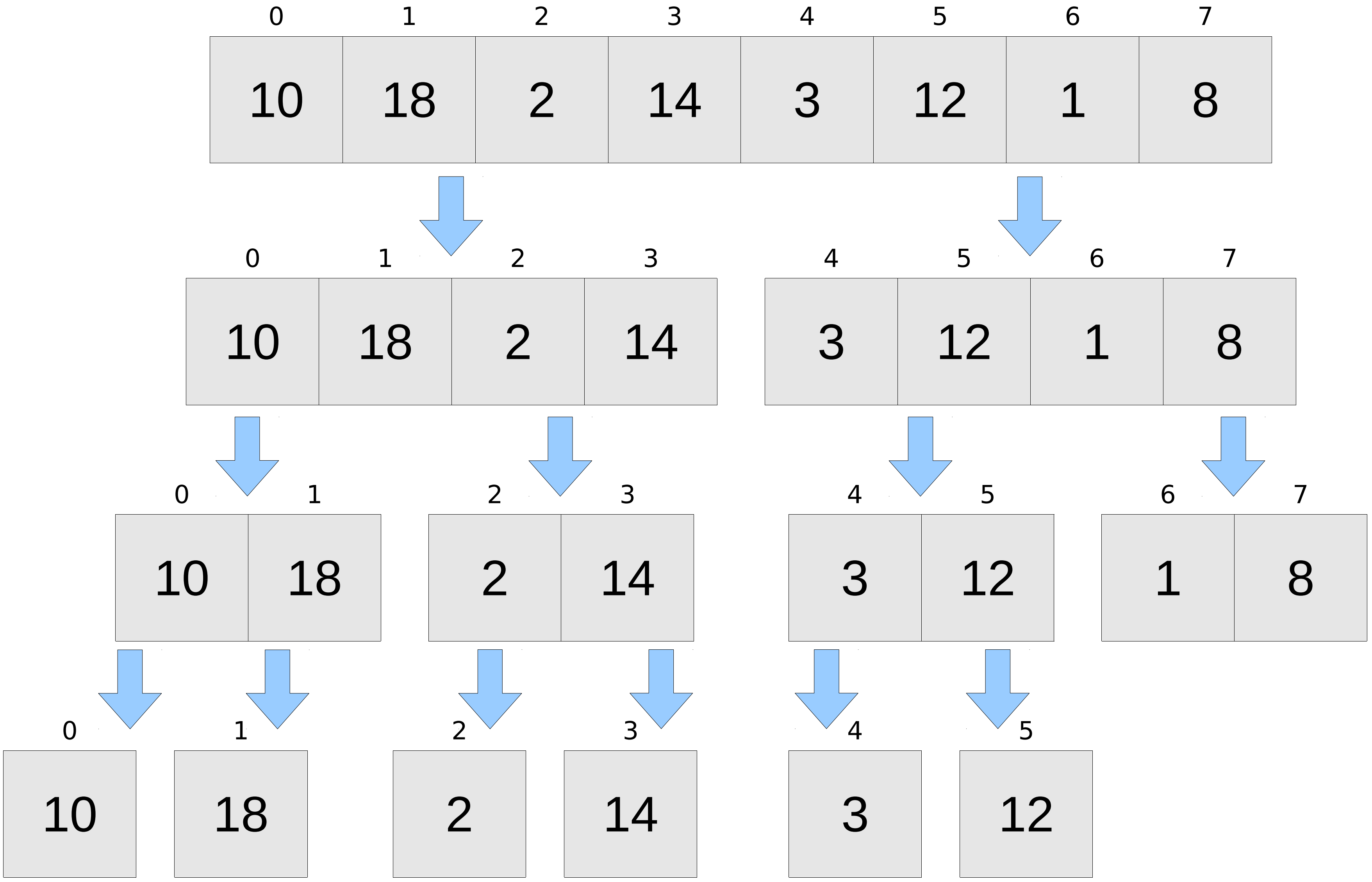
1
18

2
2

3
14

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

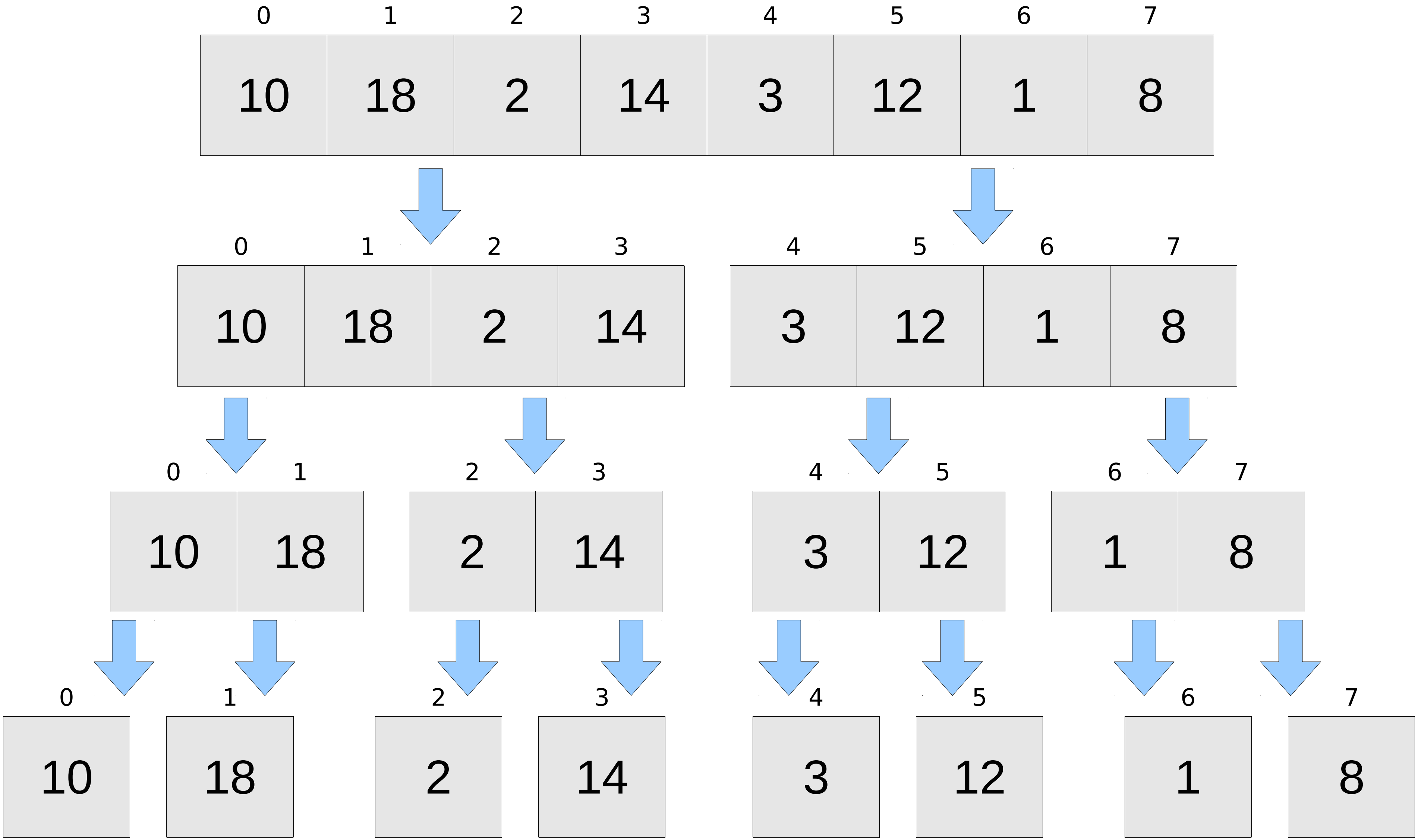
3
14

4
3

5
12

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

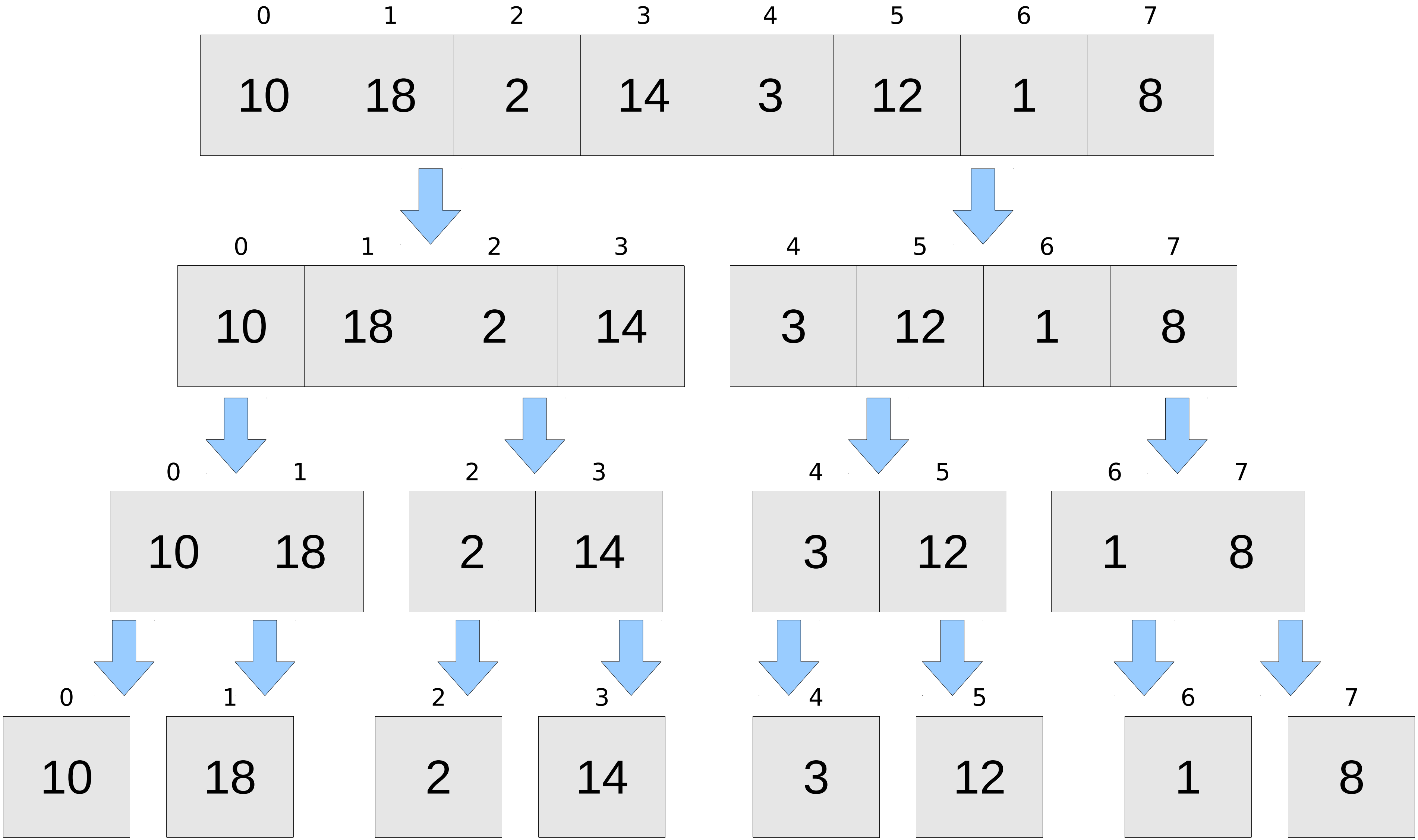
5
12

6
1

7
8

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3		4	5	6	7
10	18	2	14		3	12	1	8

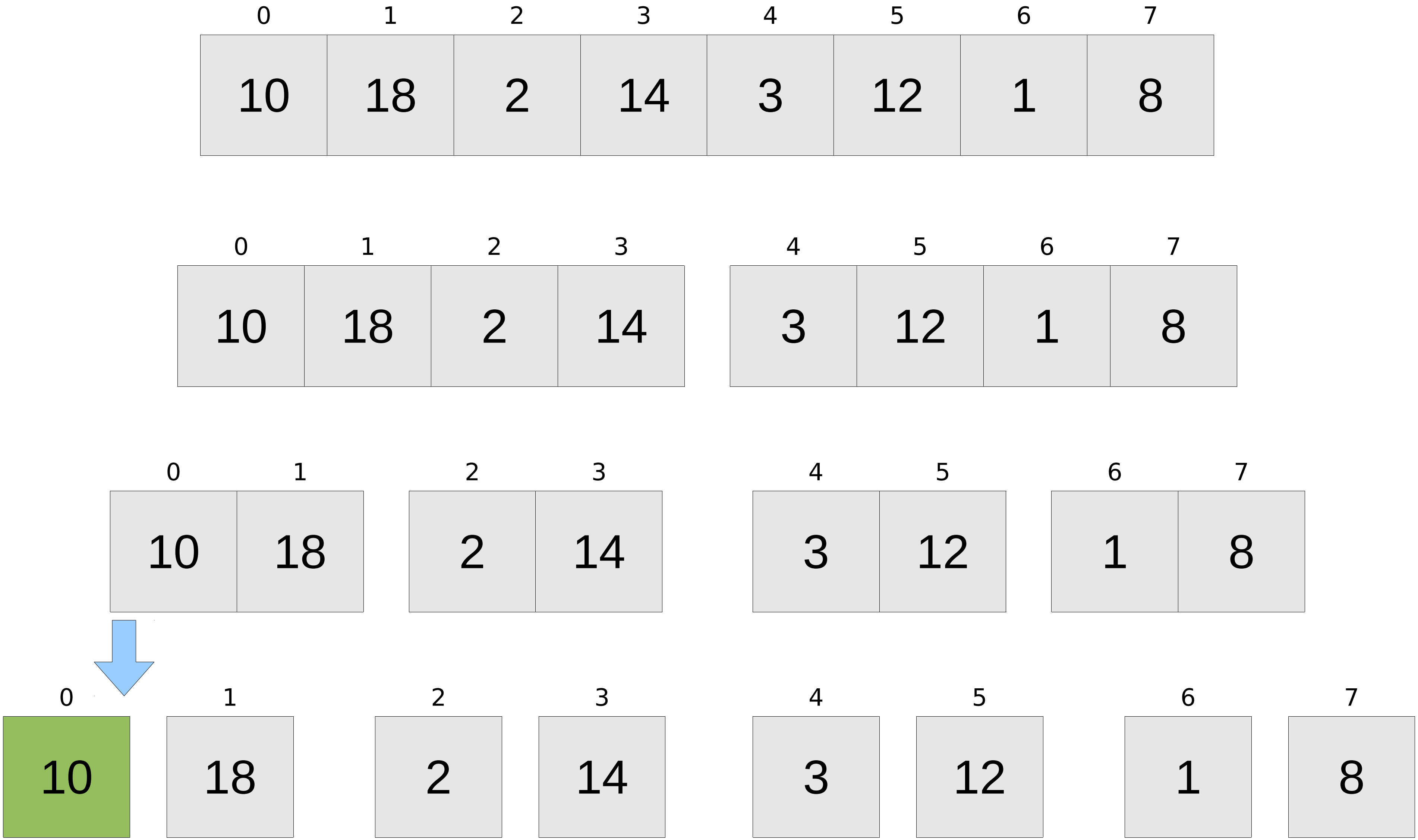
0	1		2	3		4	5		6	7
10	18		2	14		3	12		1	8

0		1		2		3		4		5		6		7
10		18		2		14		3		12		1		8

We've reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

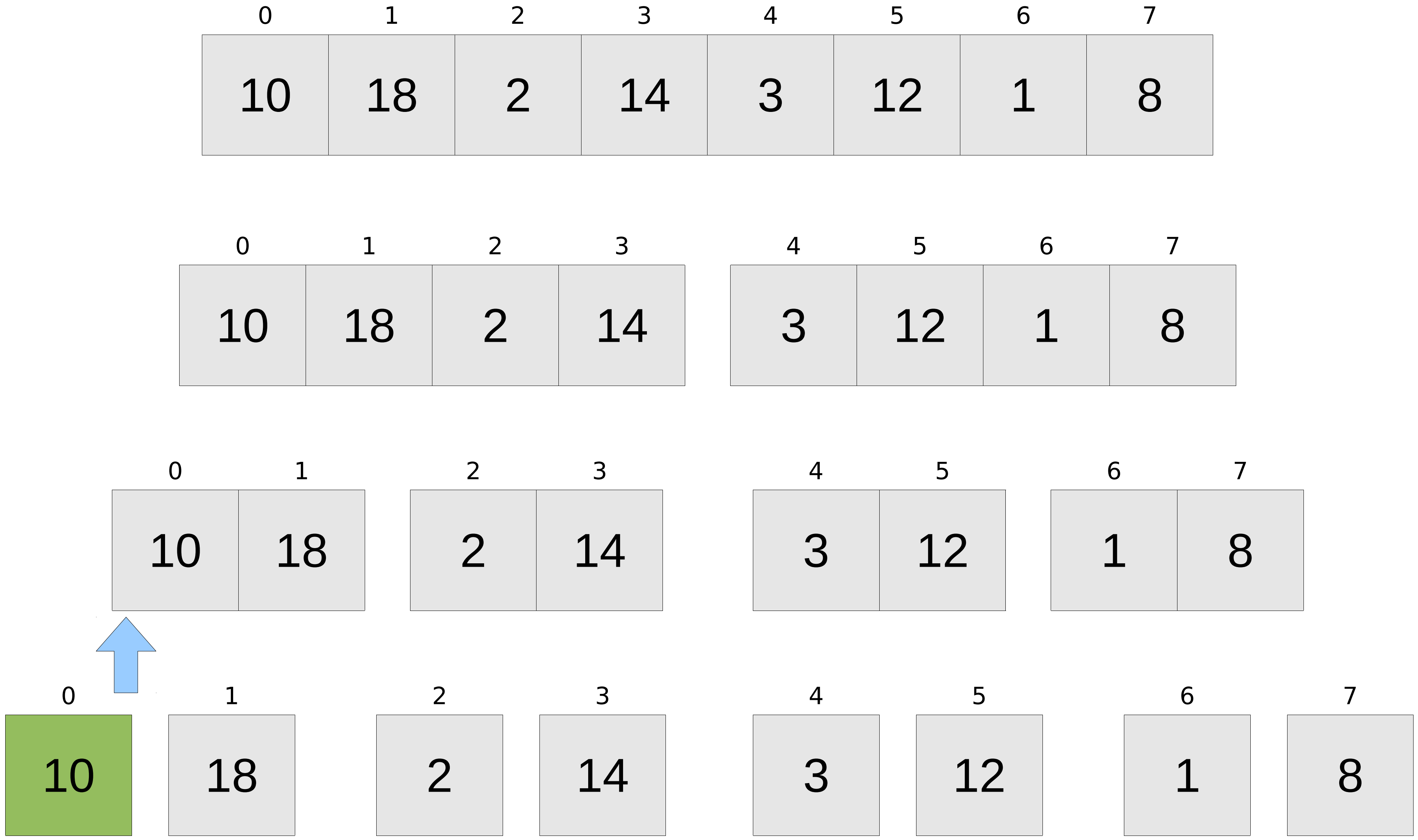
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

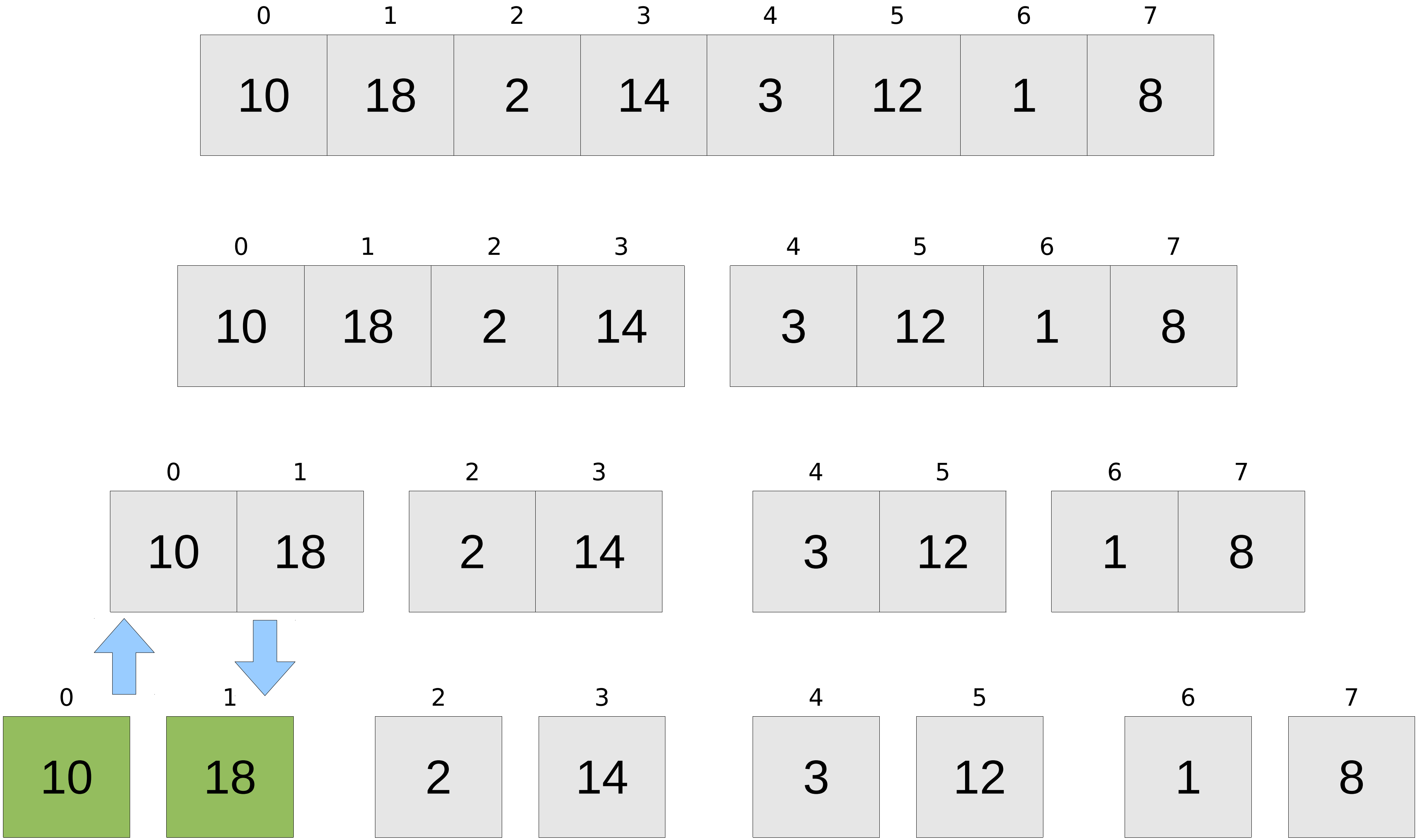
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

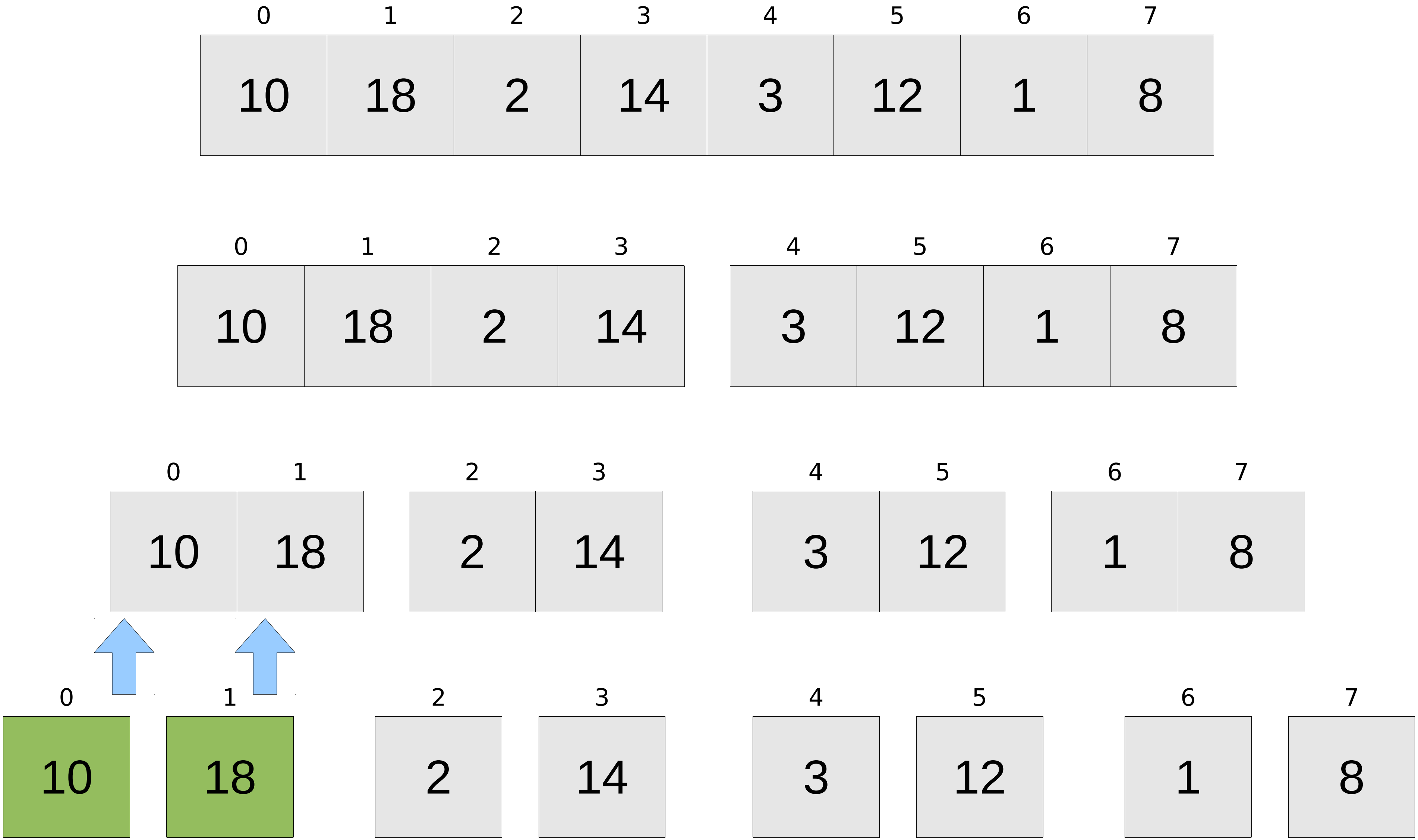
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

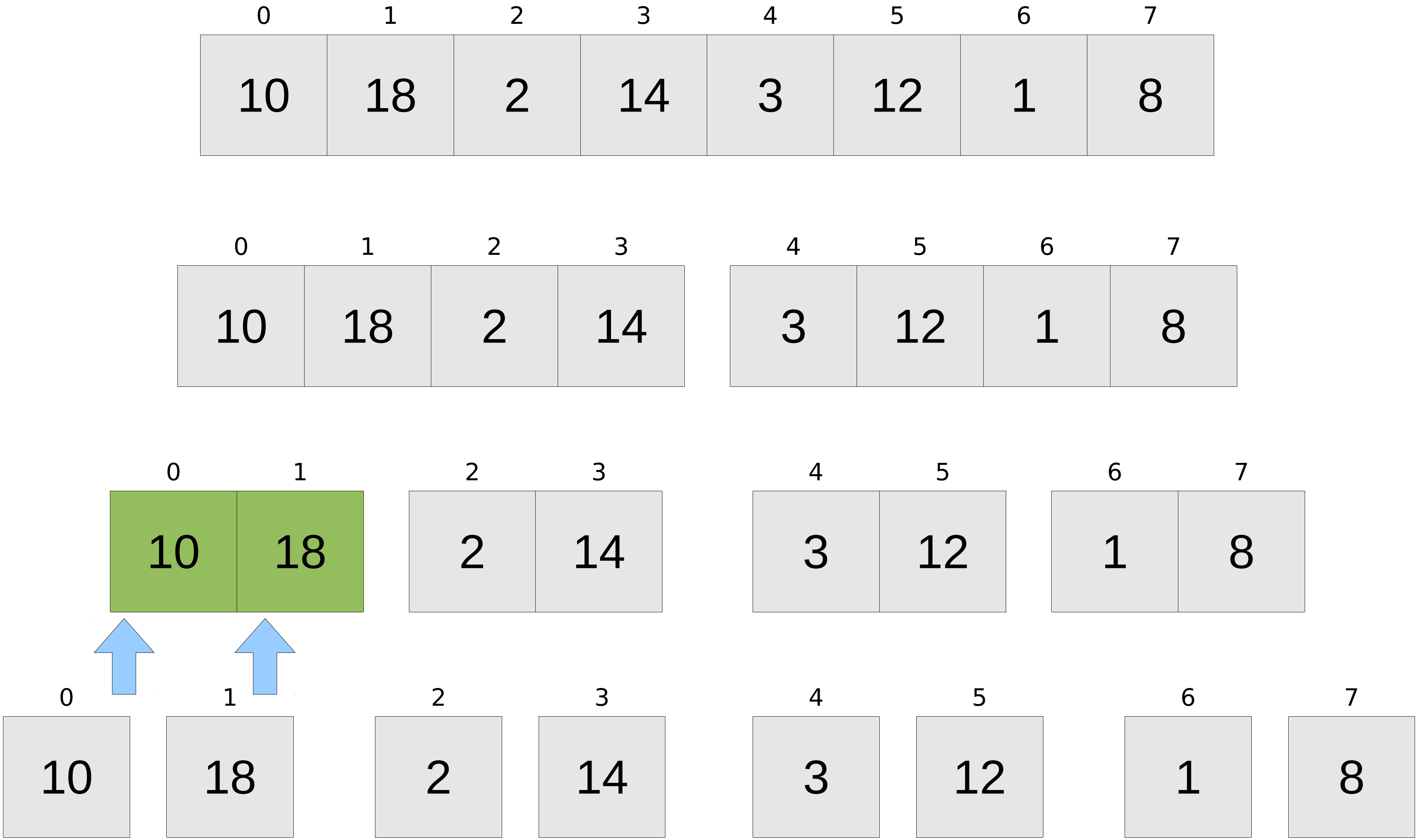
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

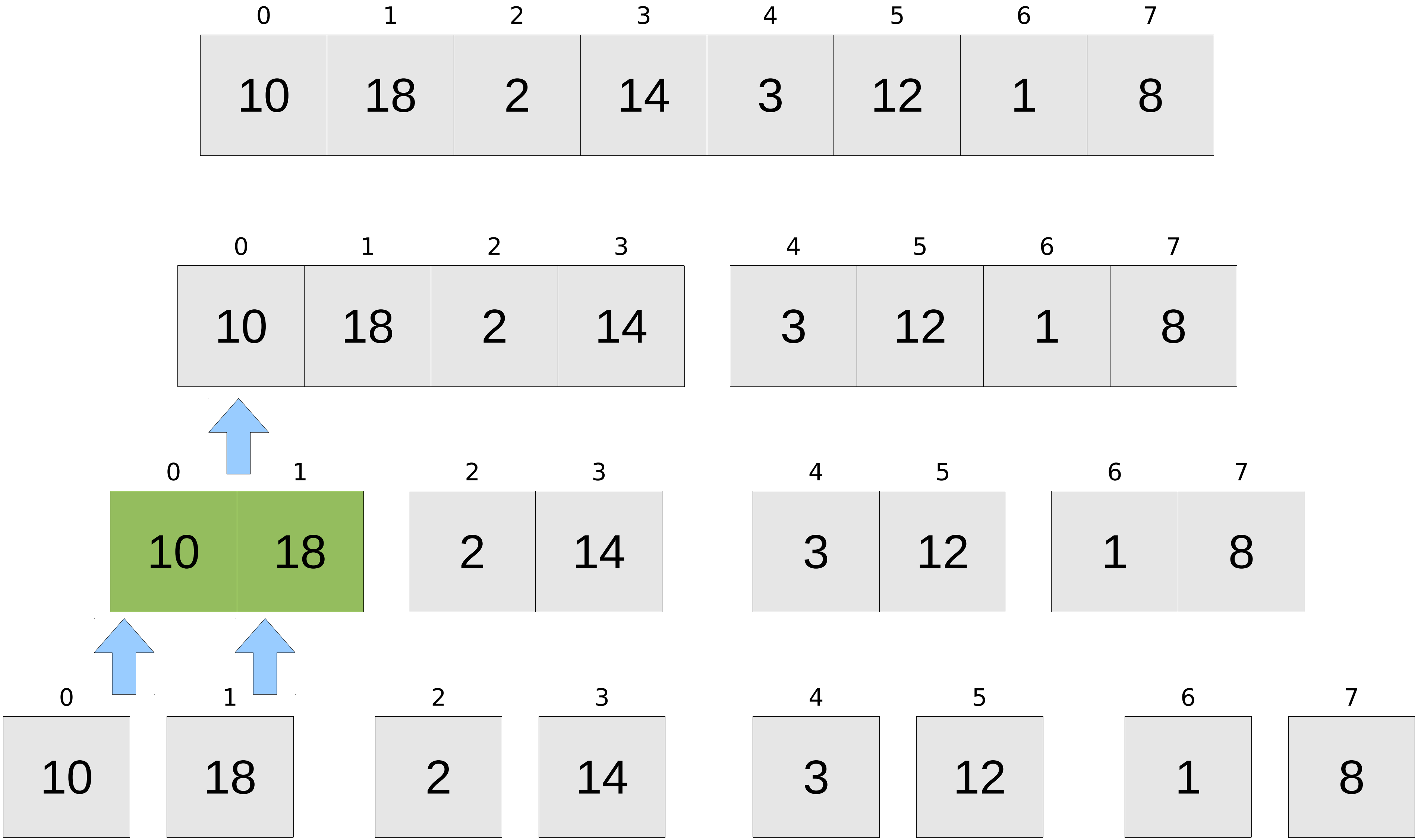
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

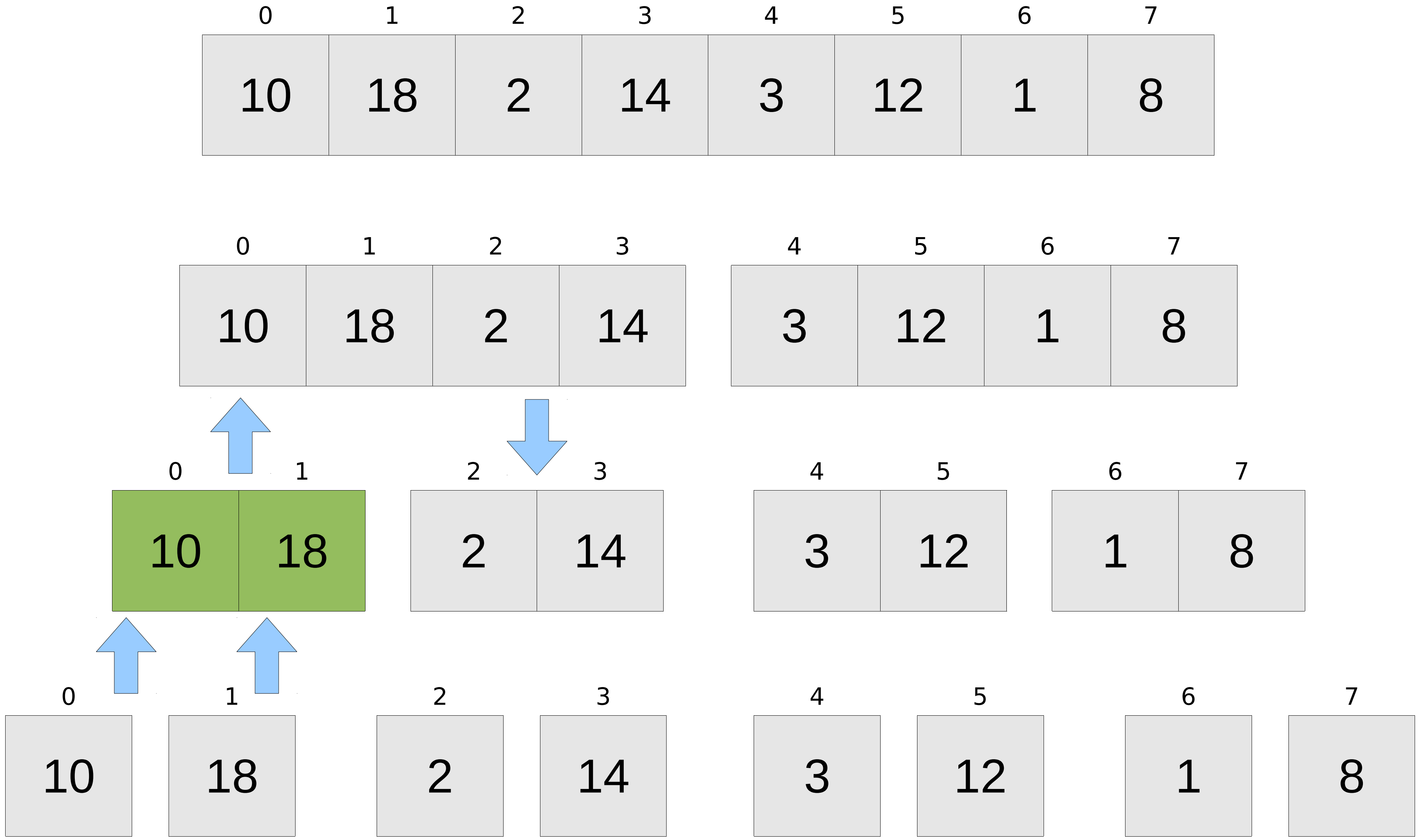
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

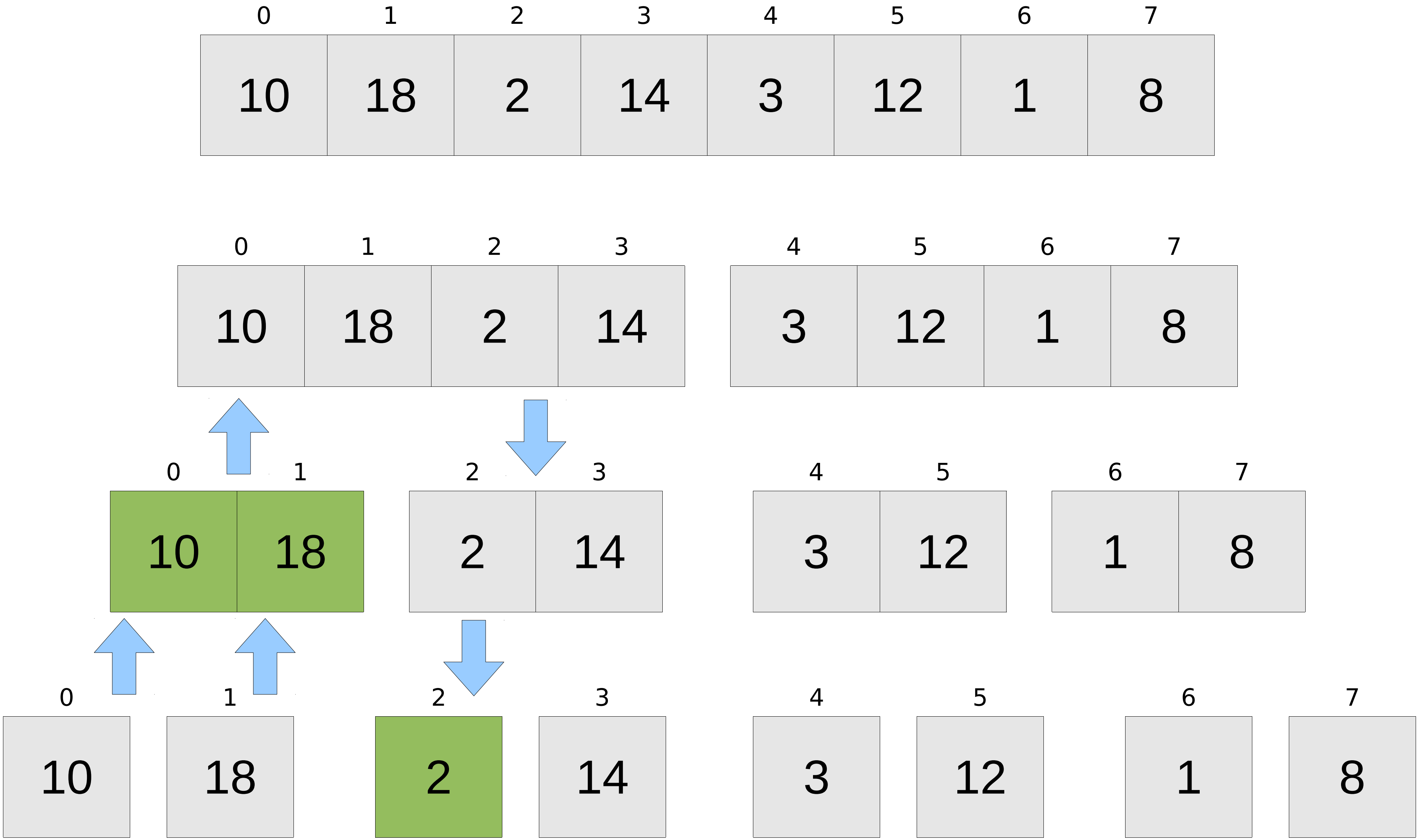
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

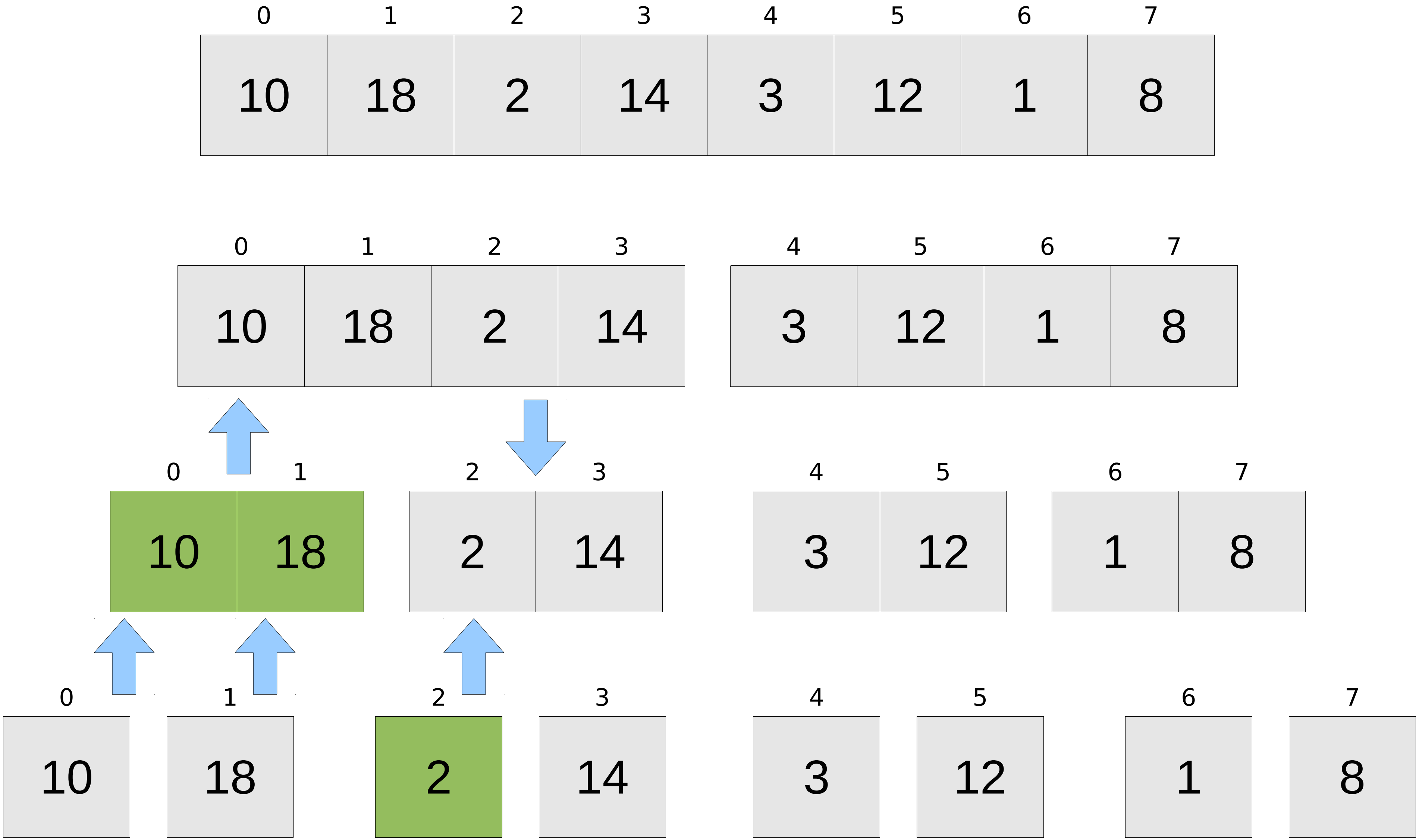
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

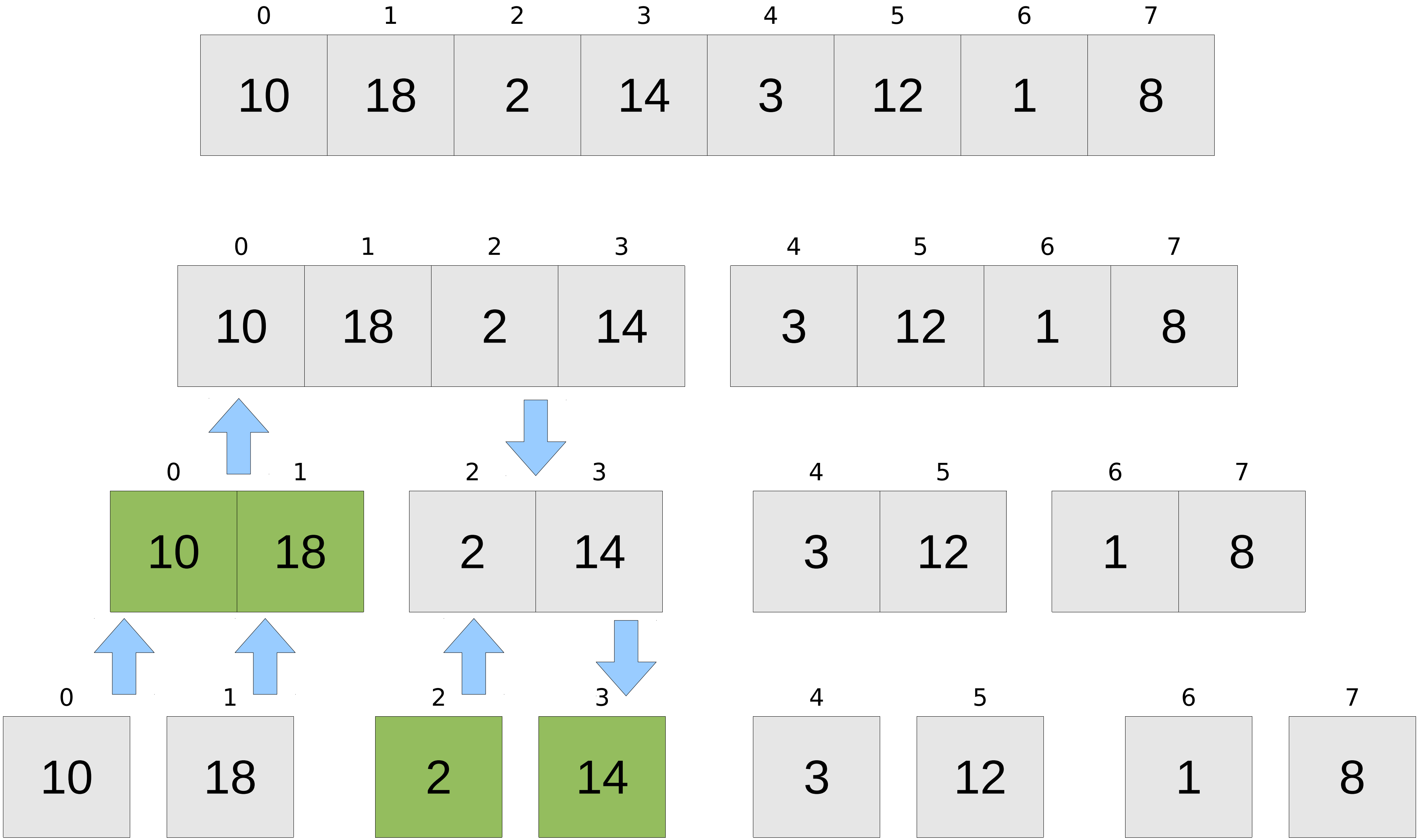
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

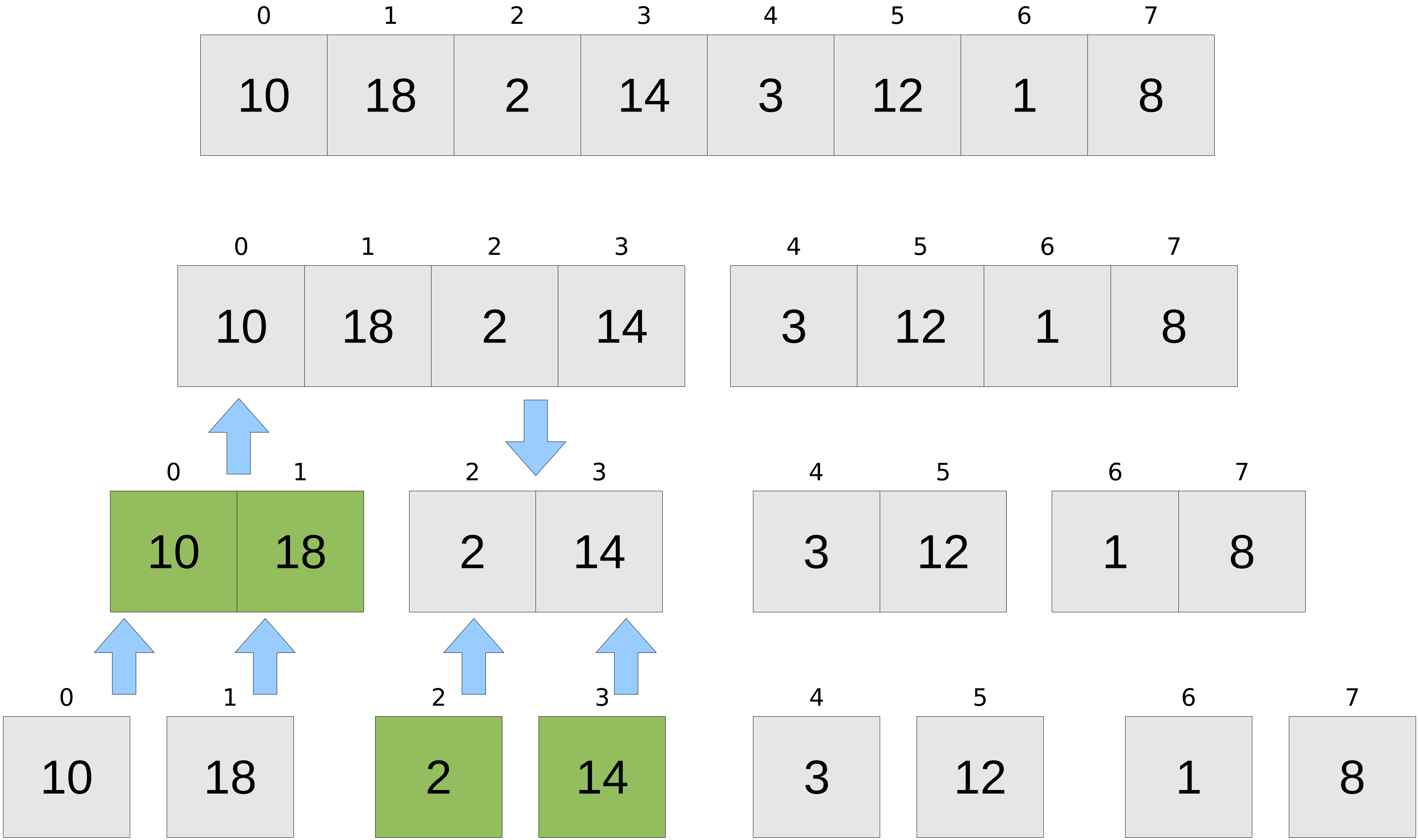
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

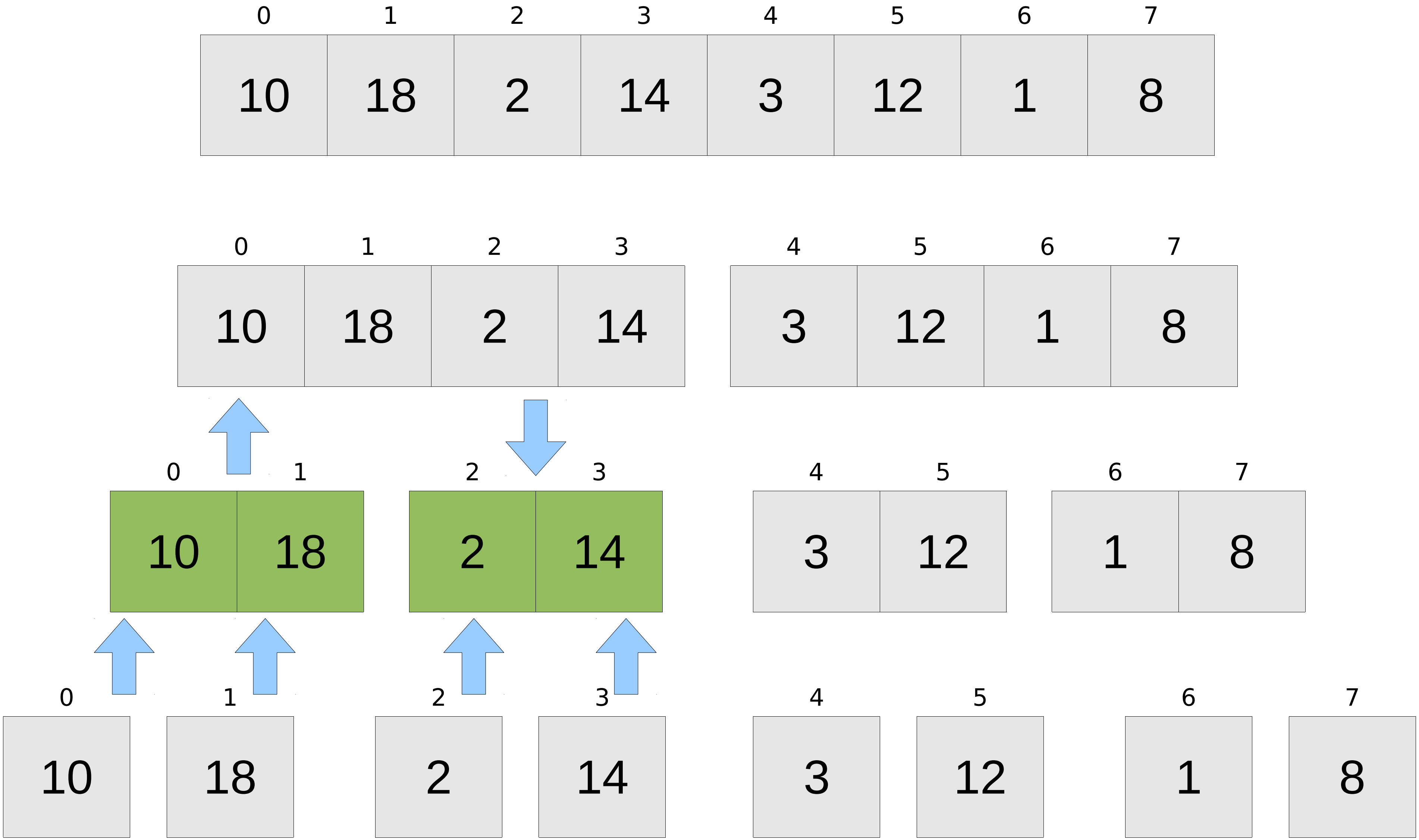
(break down the array, then merge the pieces back together)



We've reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

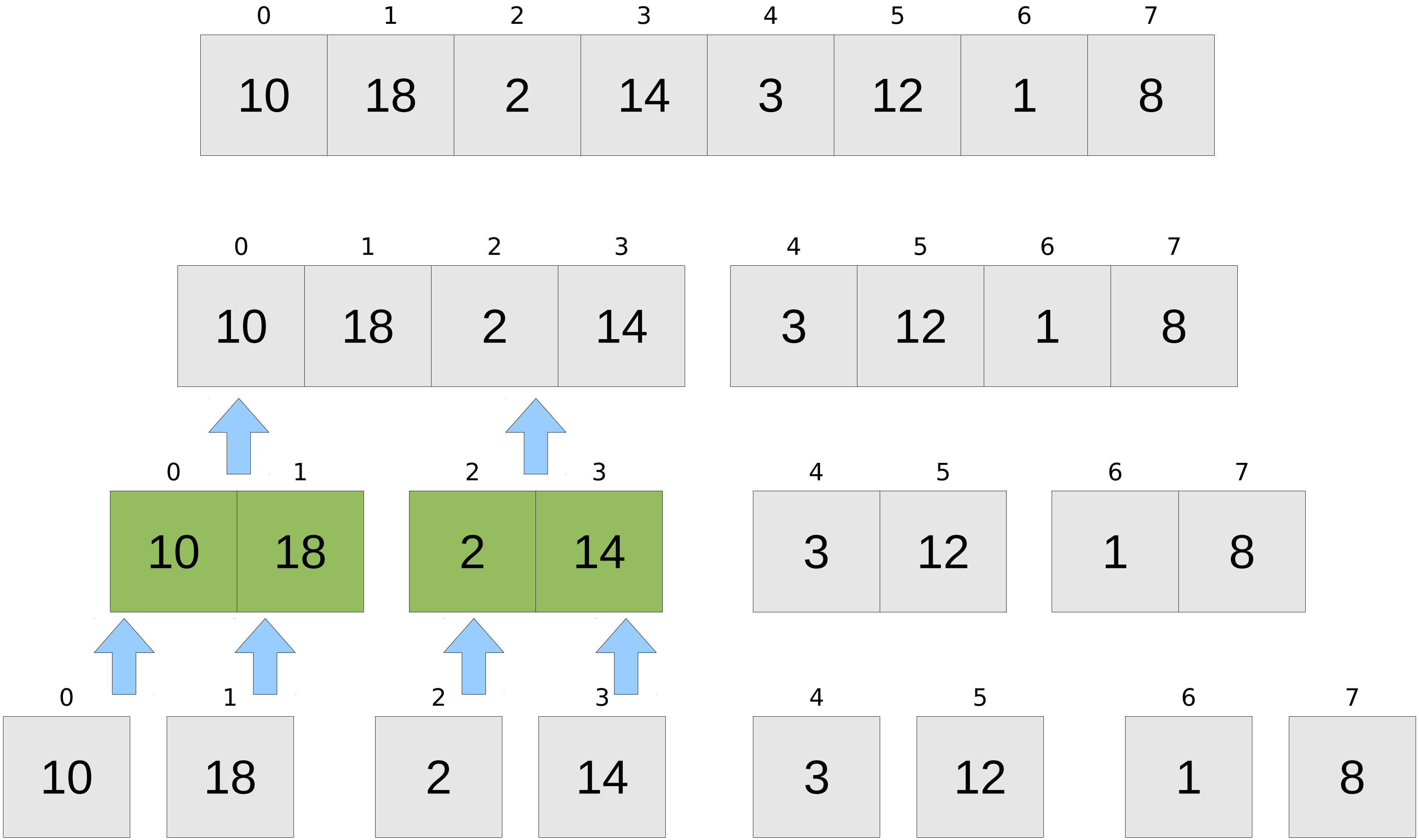
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
10	18	2	14

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

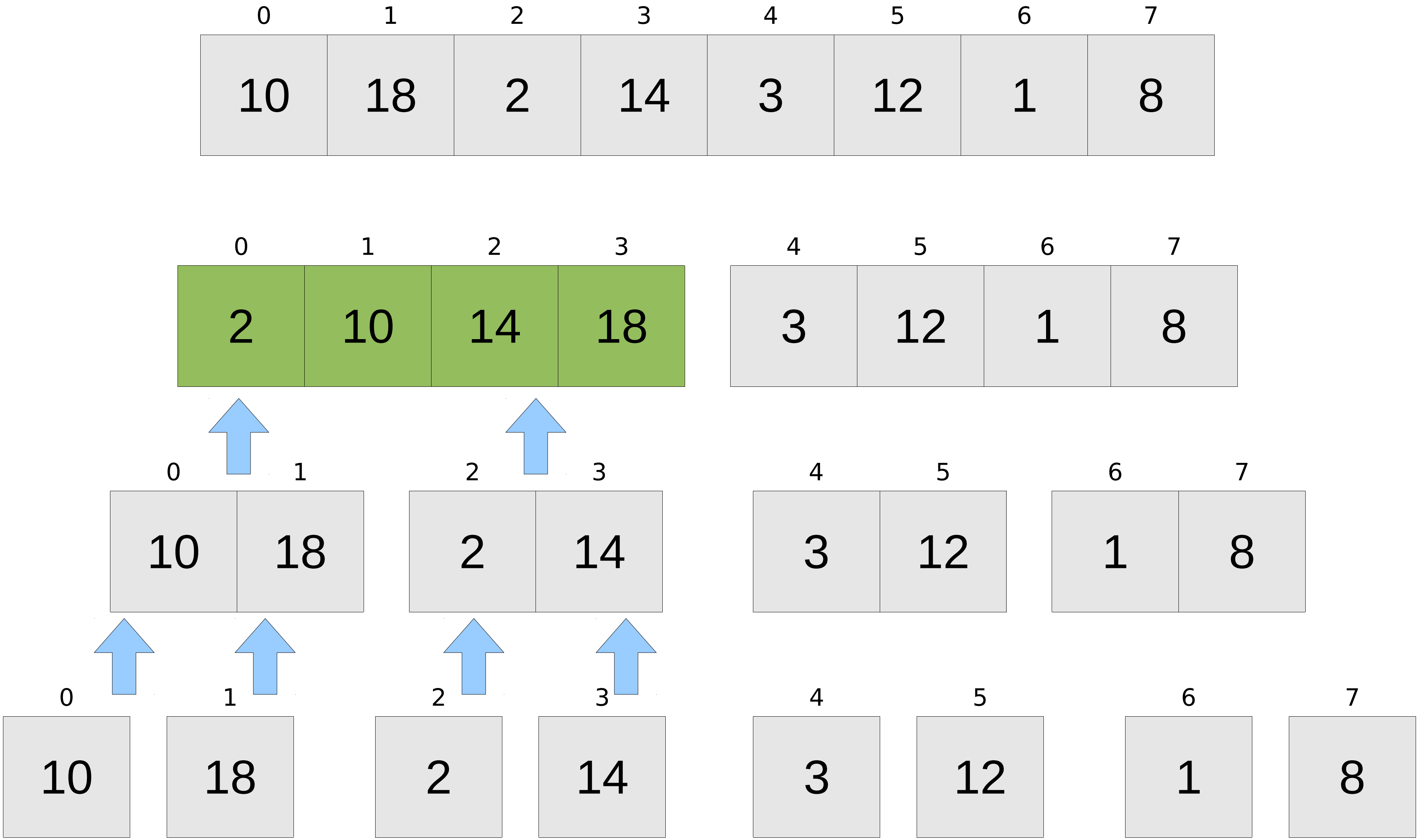
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

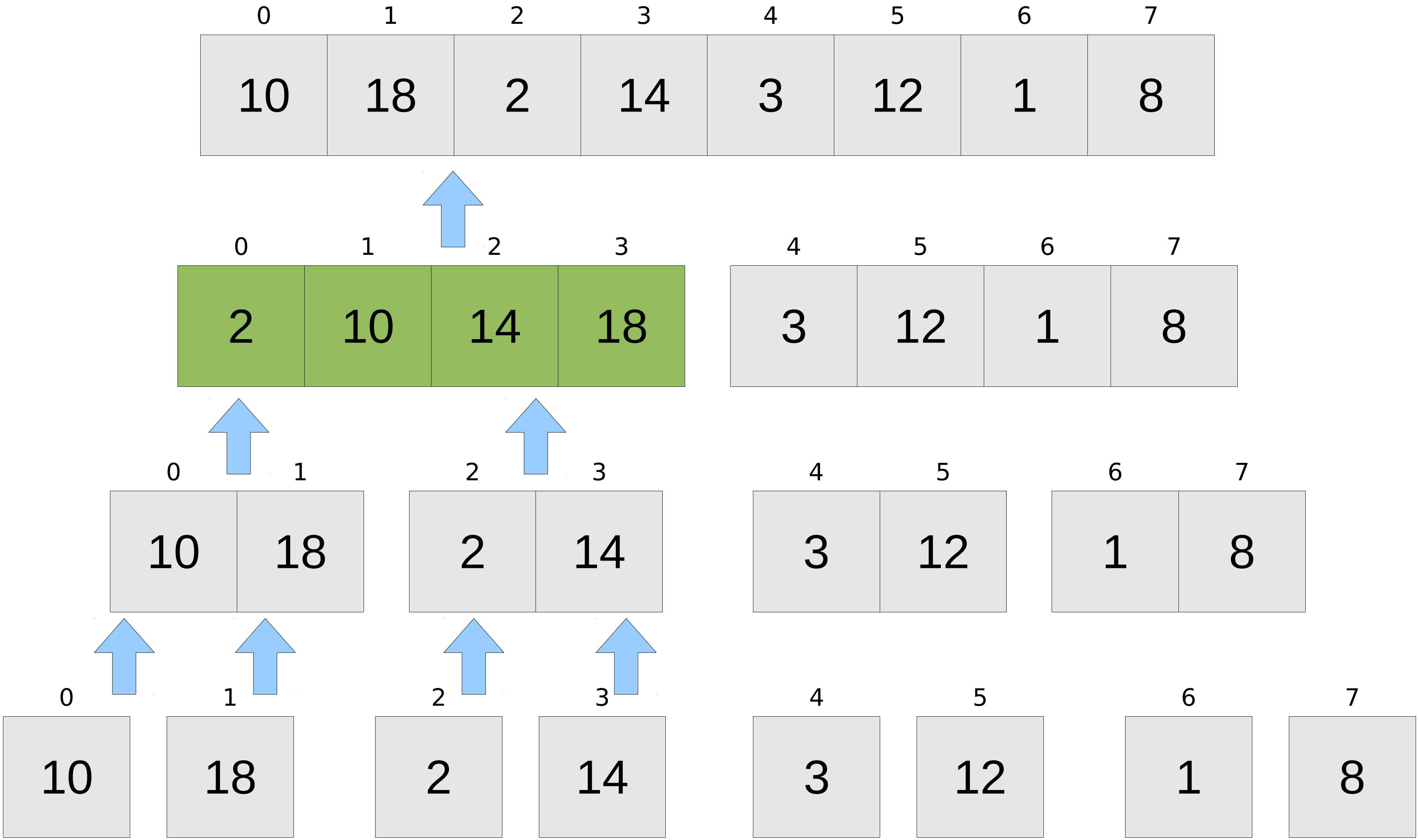
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

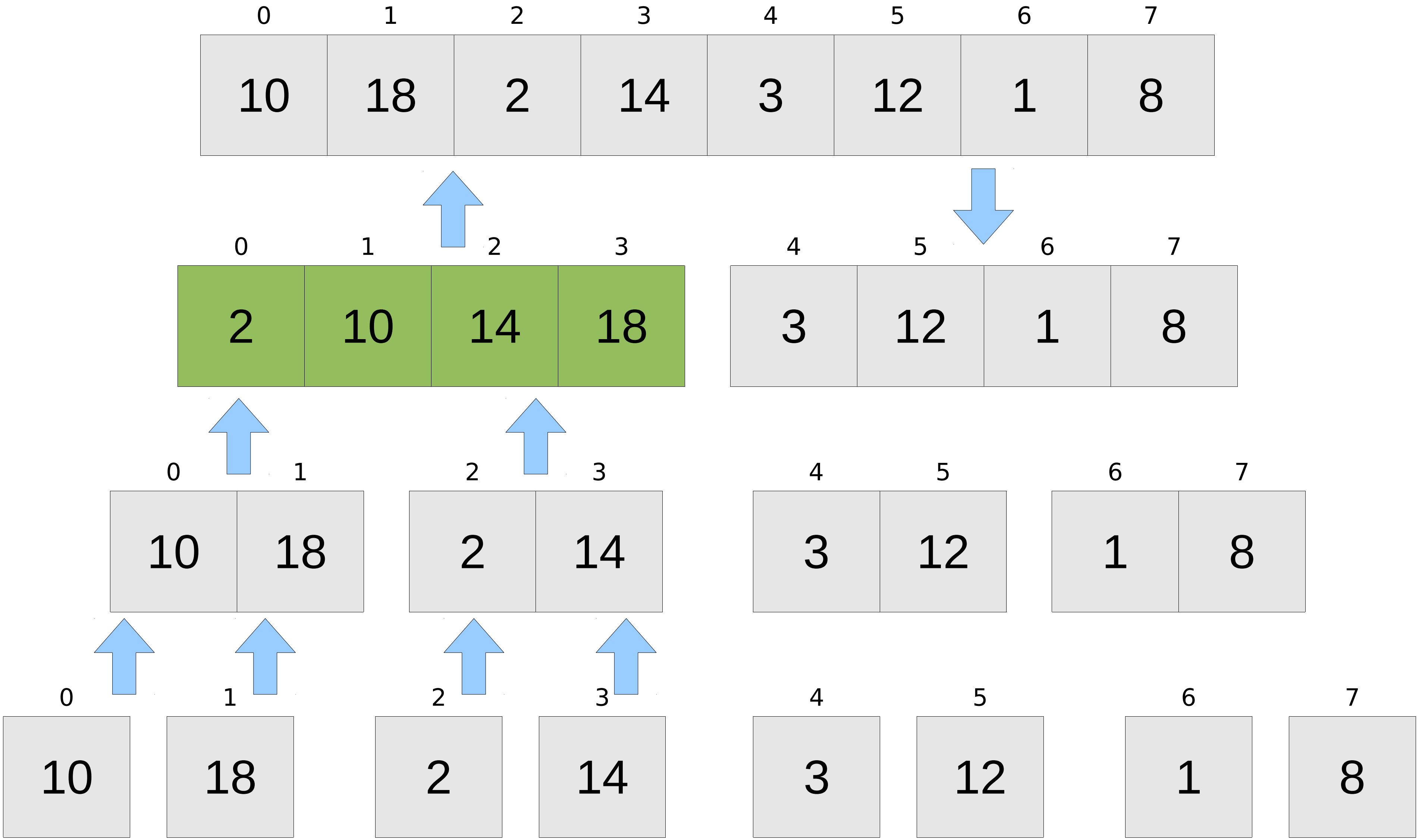
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

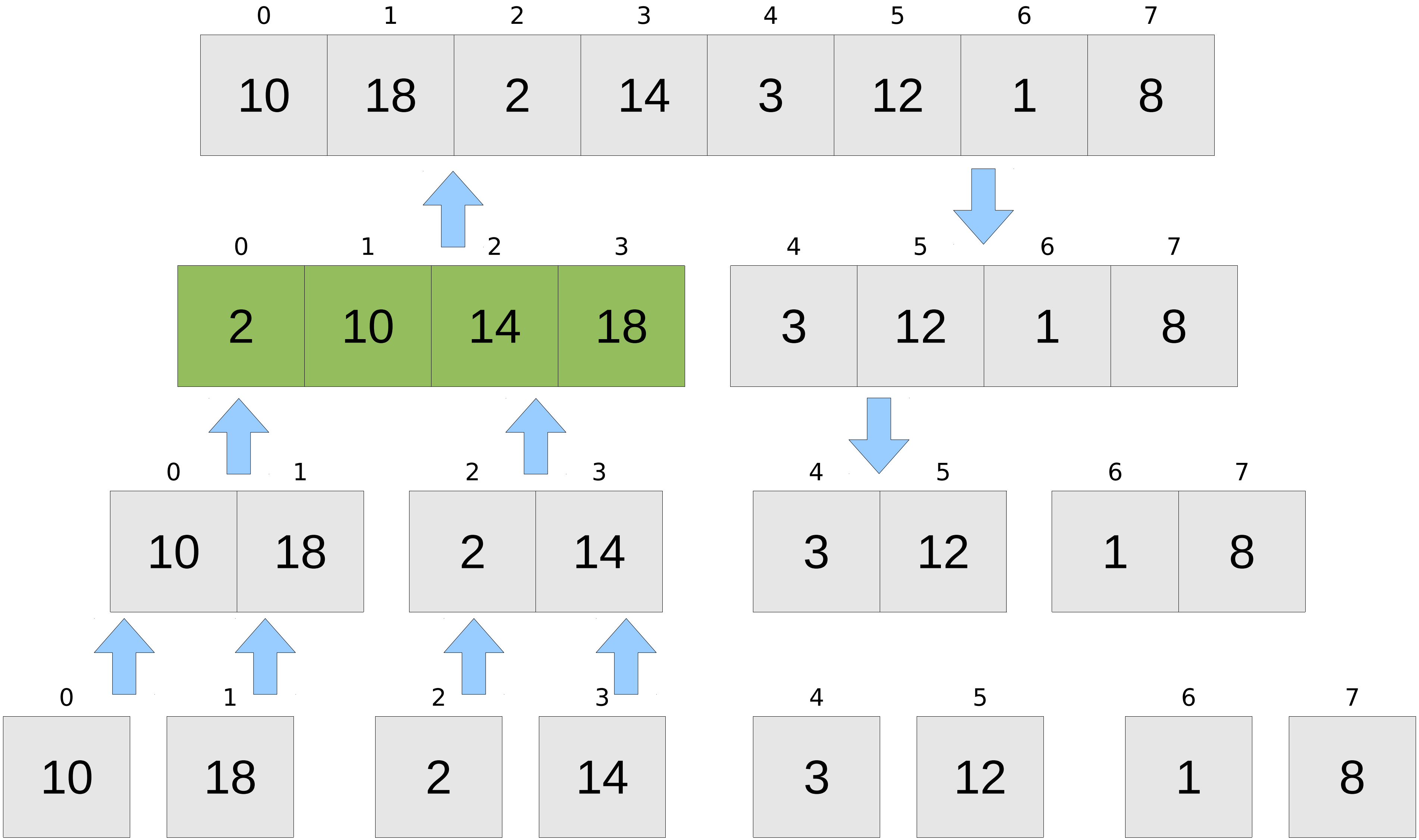
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

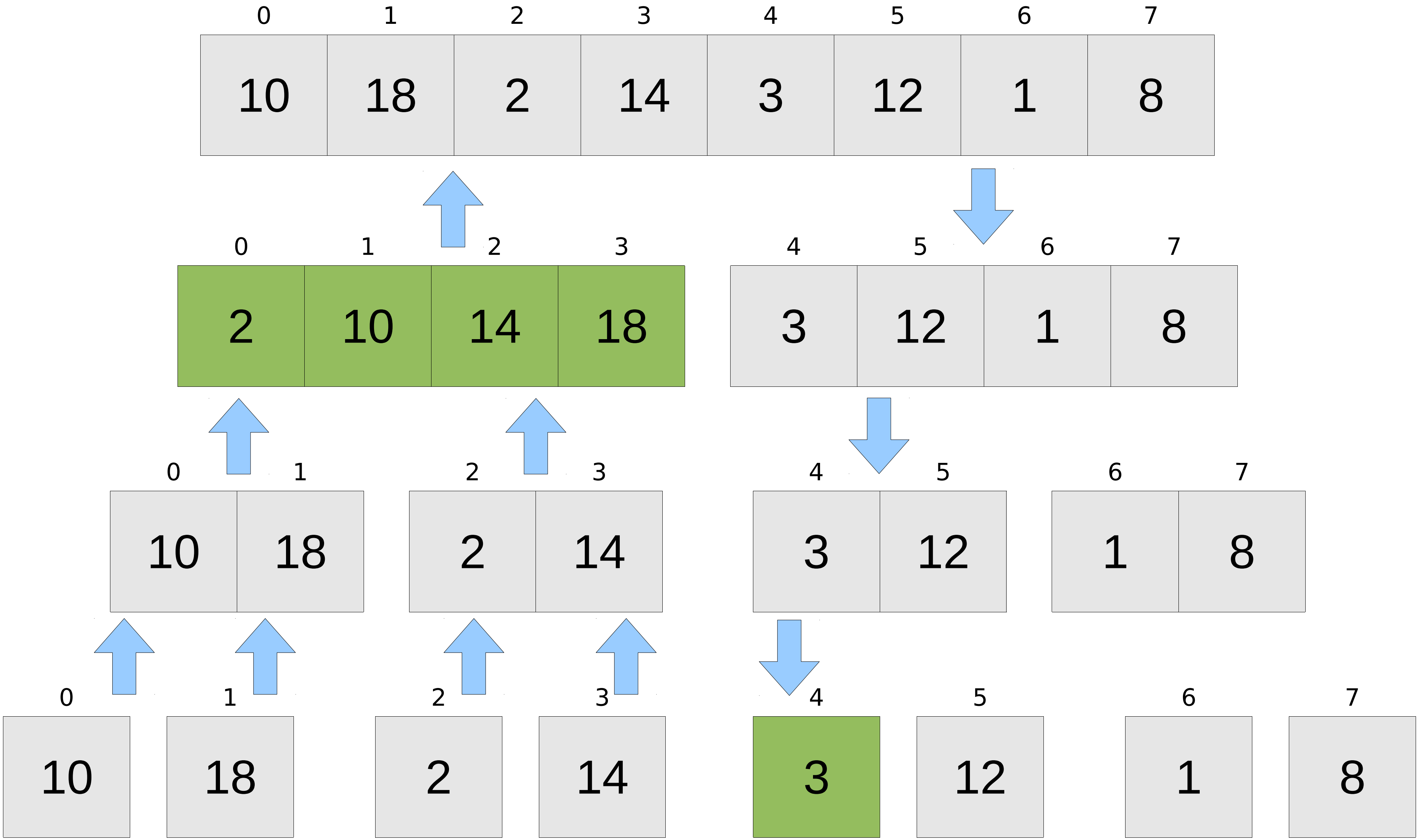
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

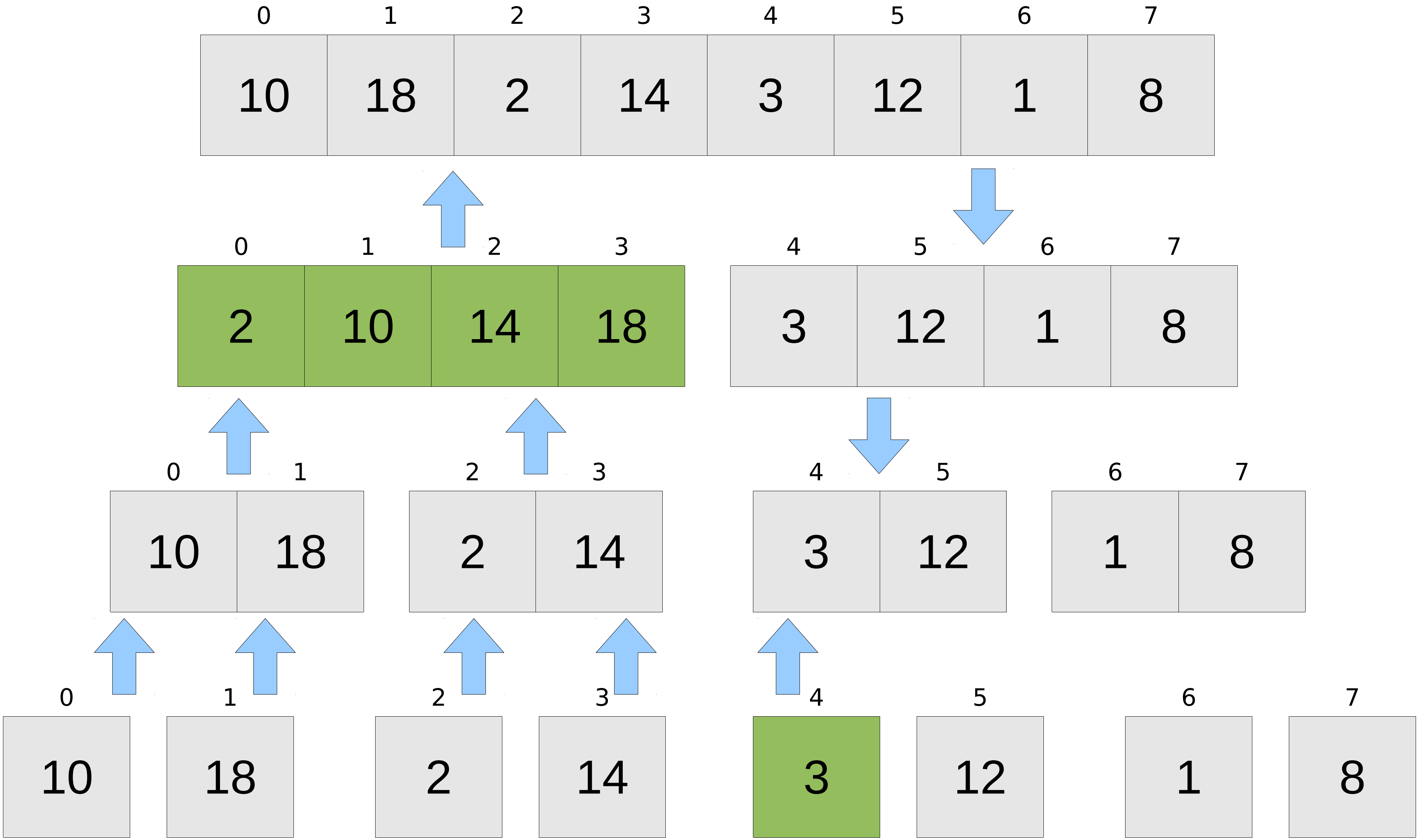
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3	4	5	6	7
2	10	14	18	3	12	1	8

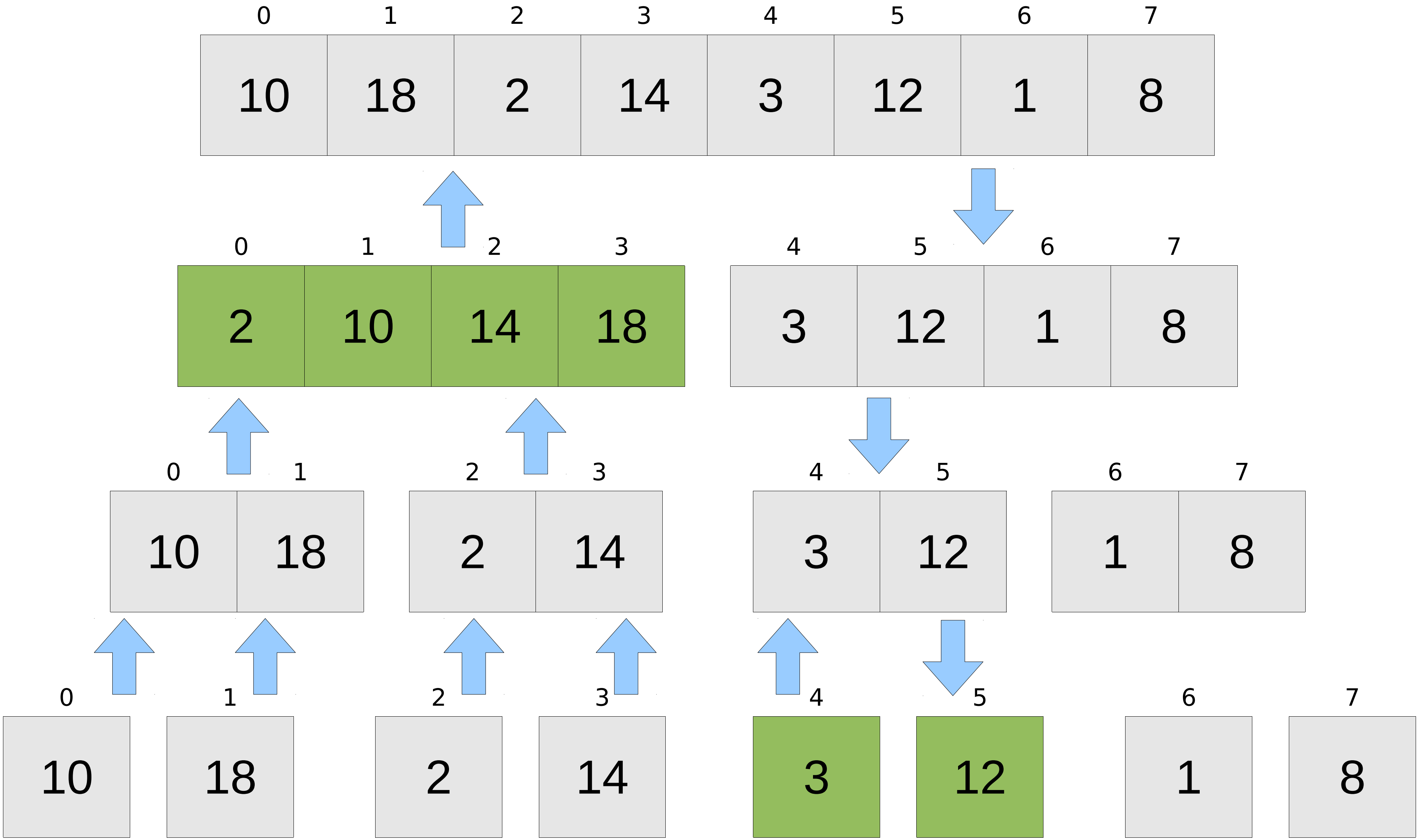
0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

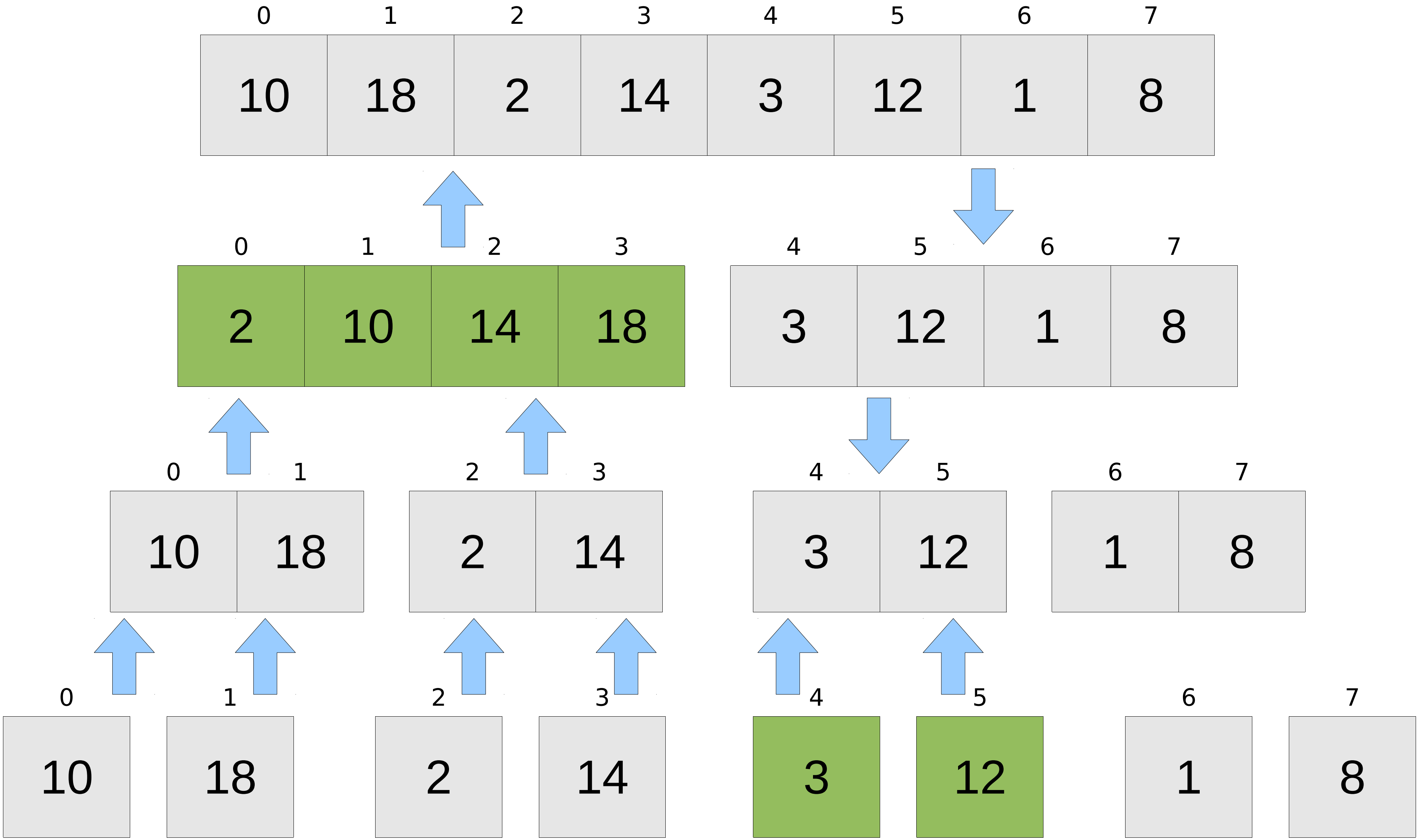
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

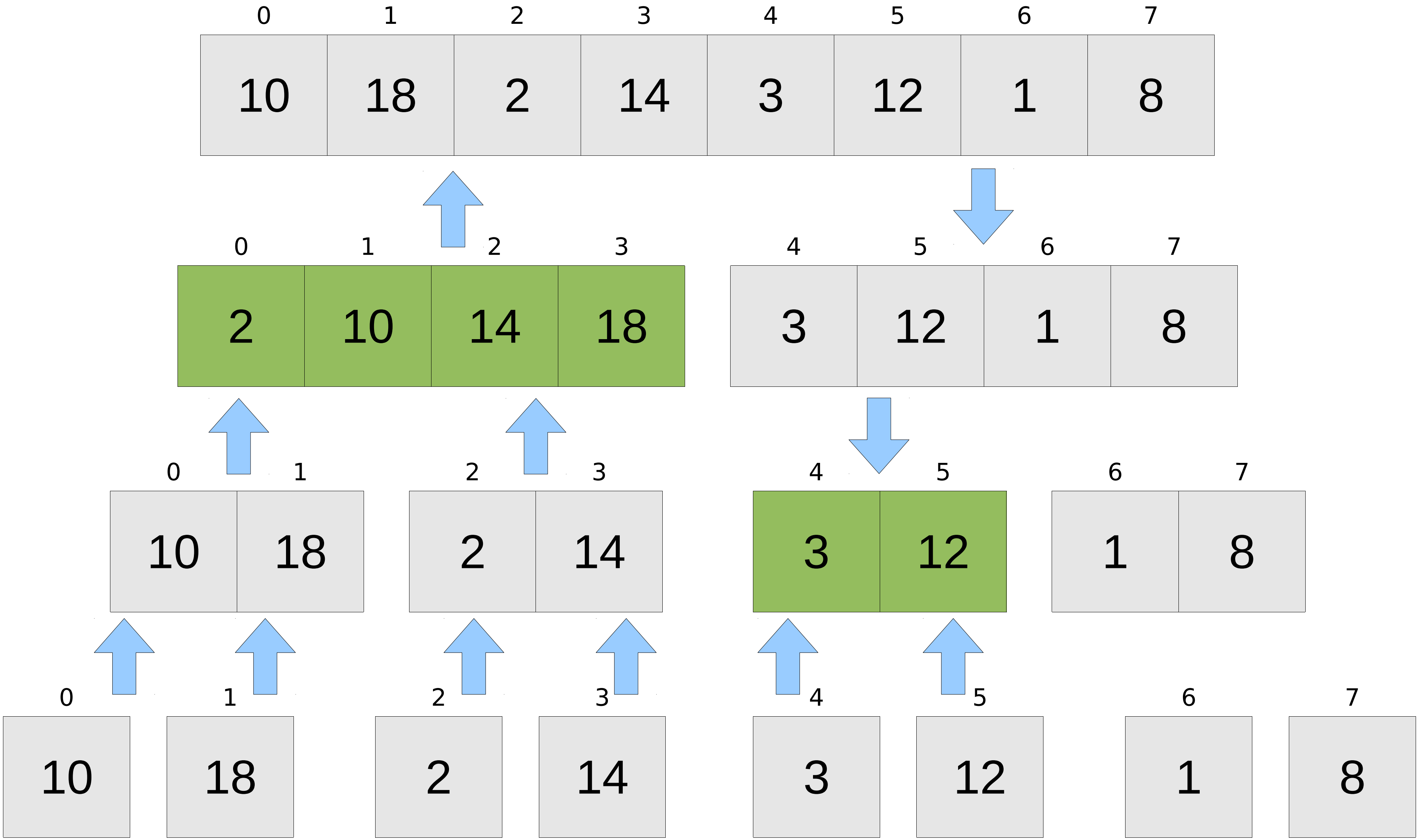
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

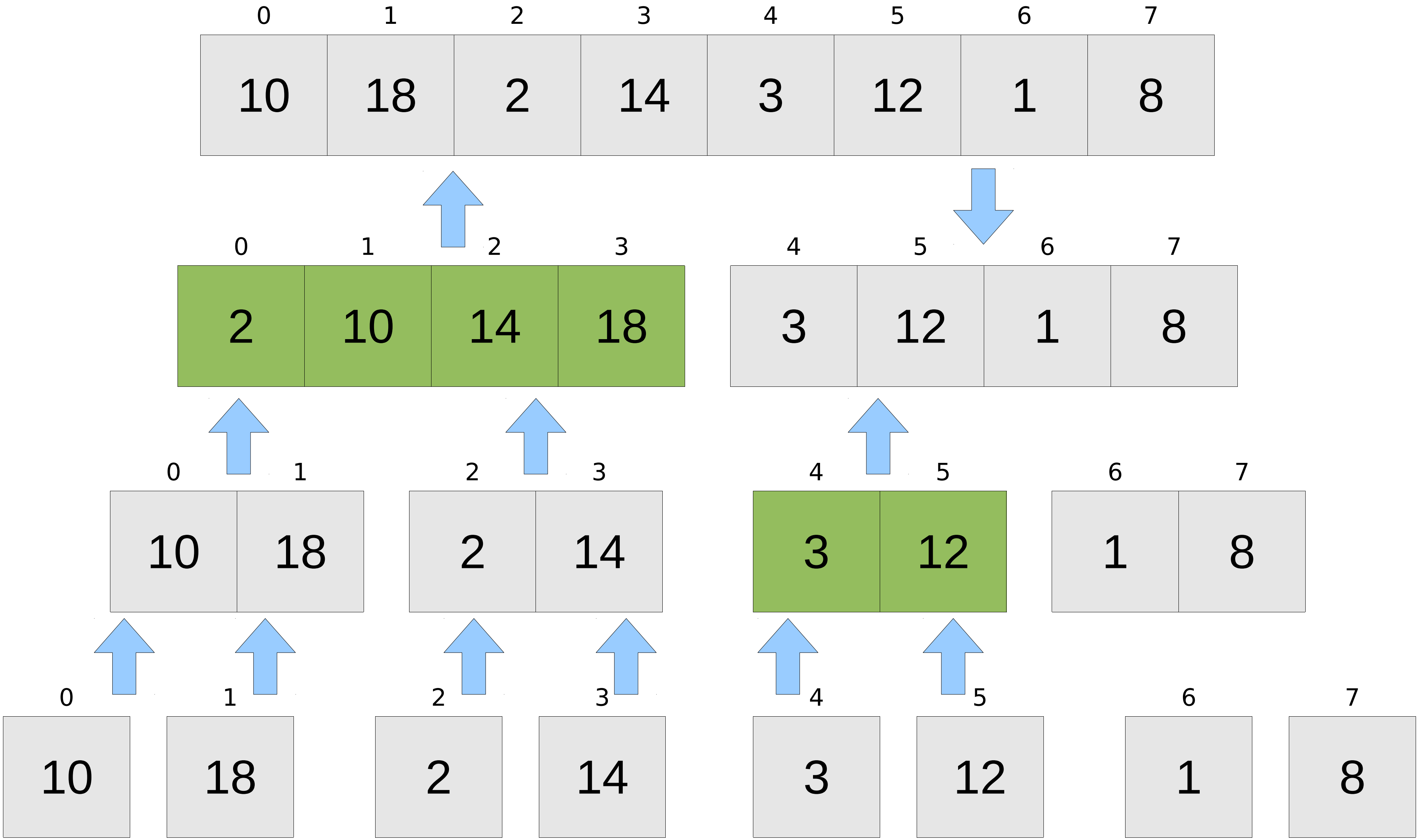
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

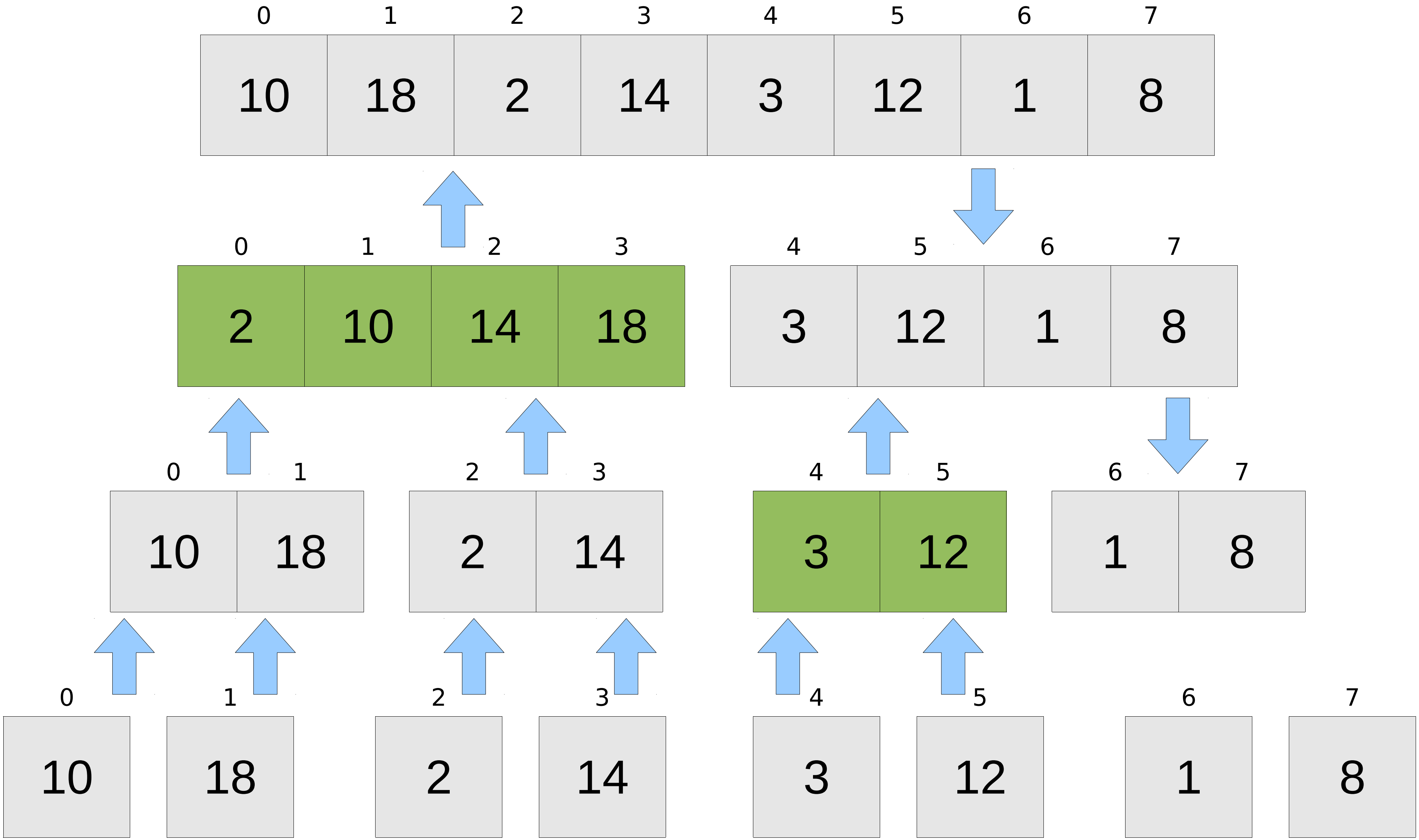
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

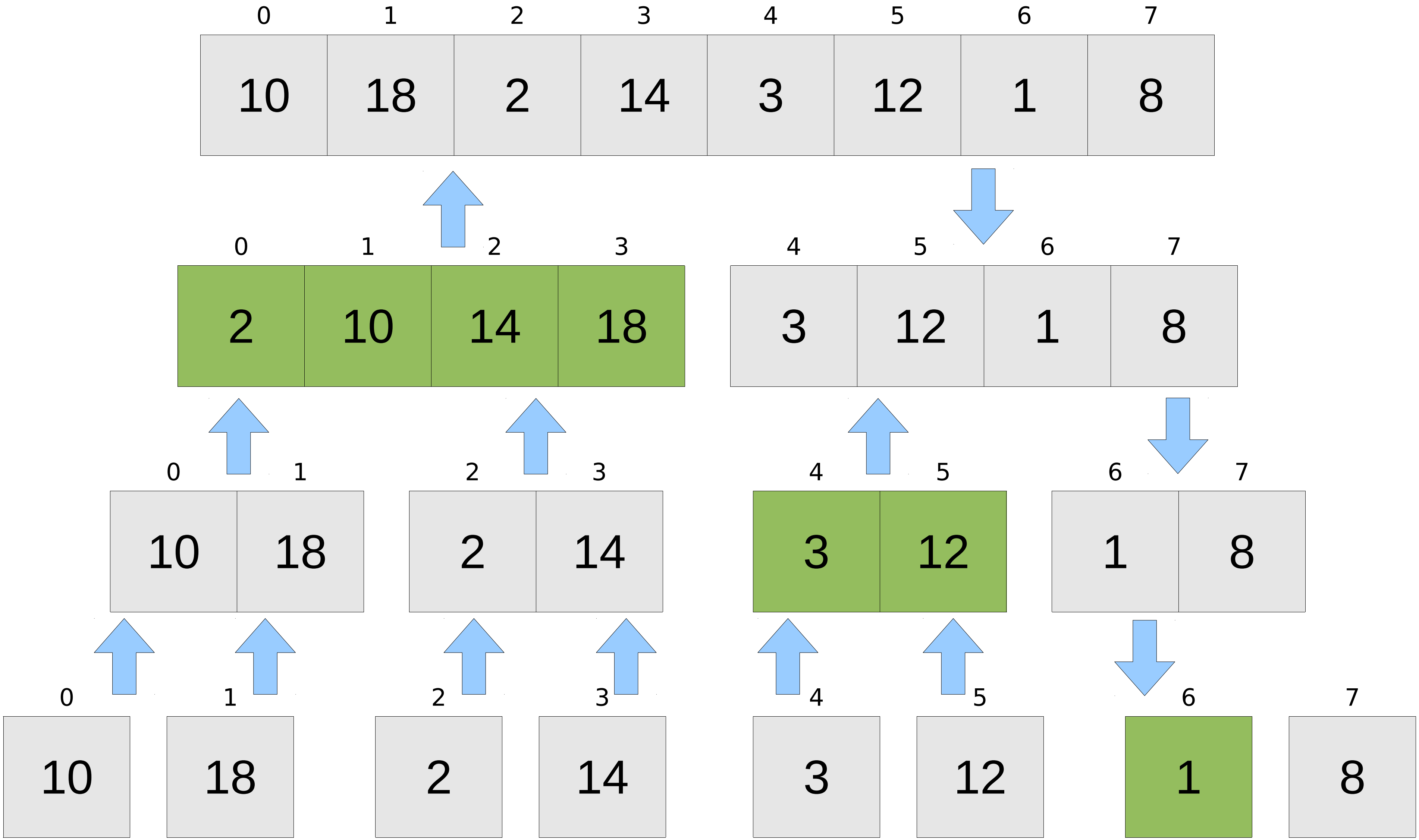
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

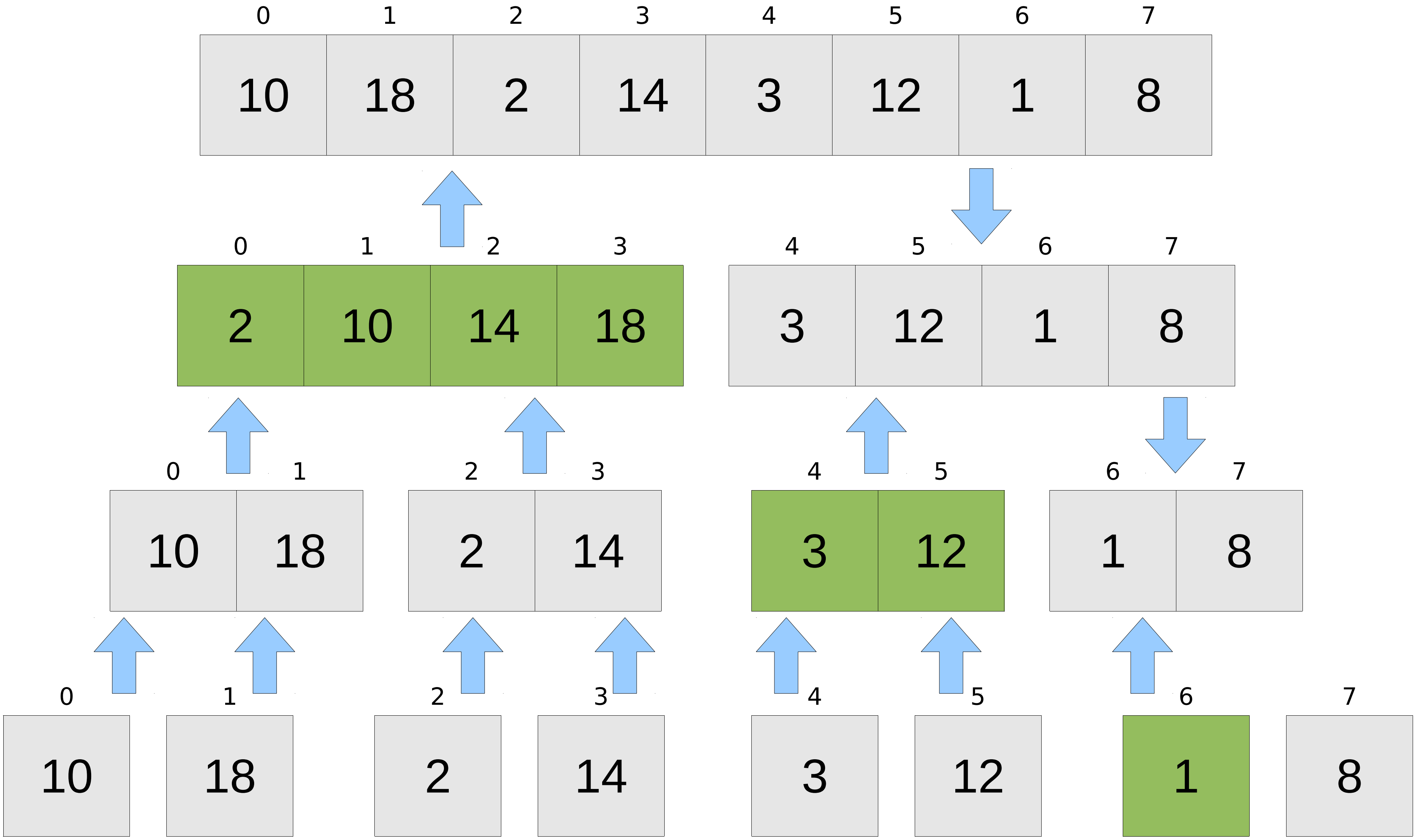
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

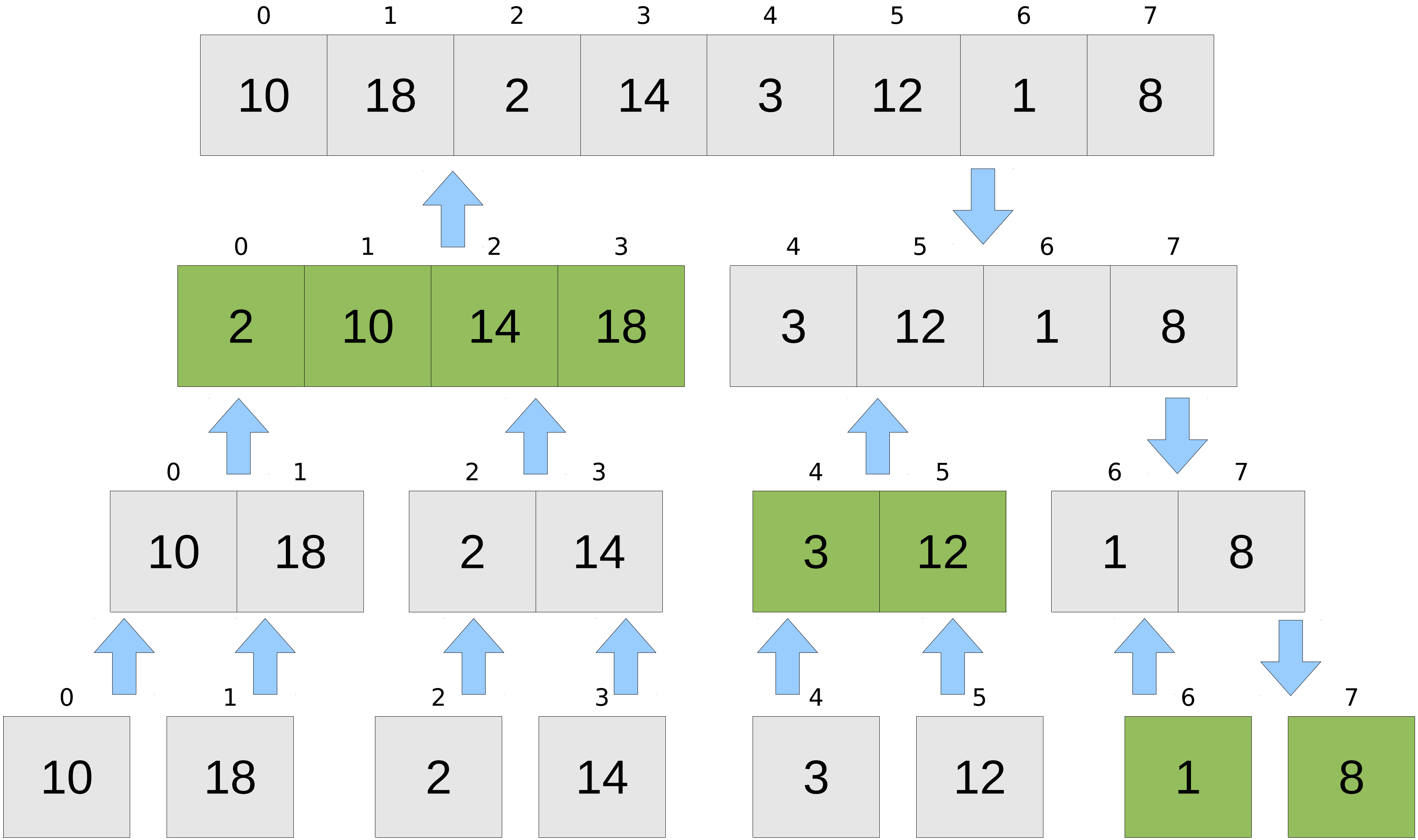
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

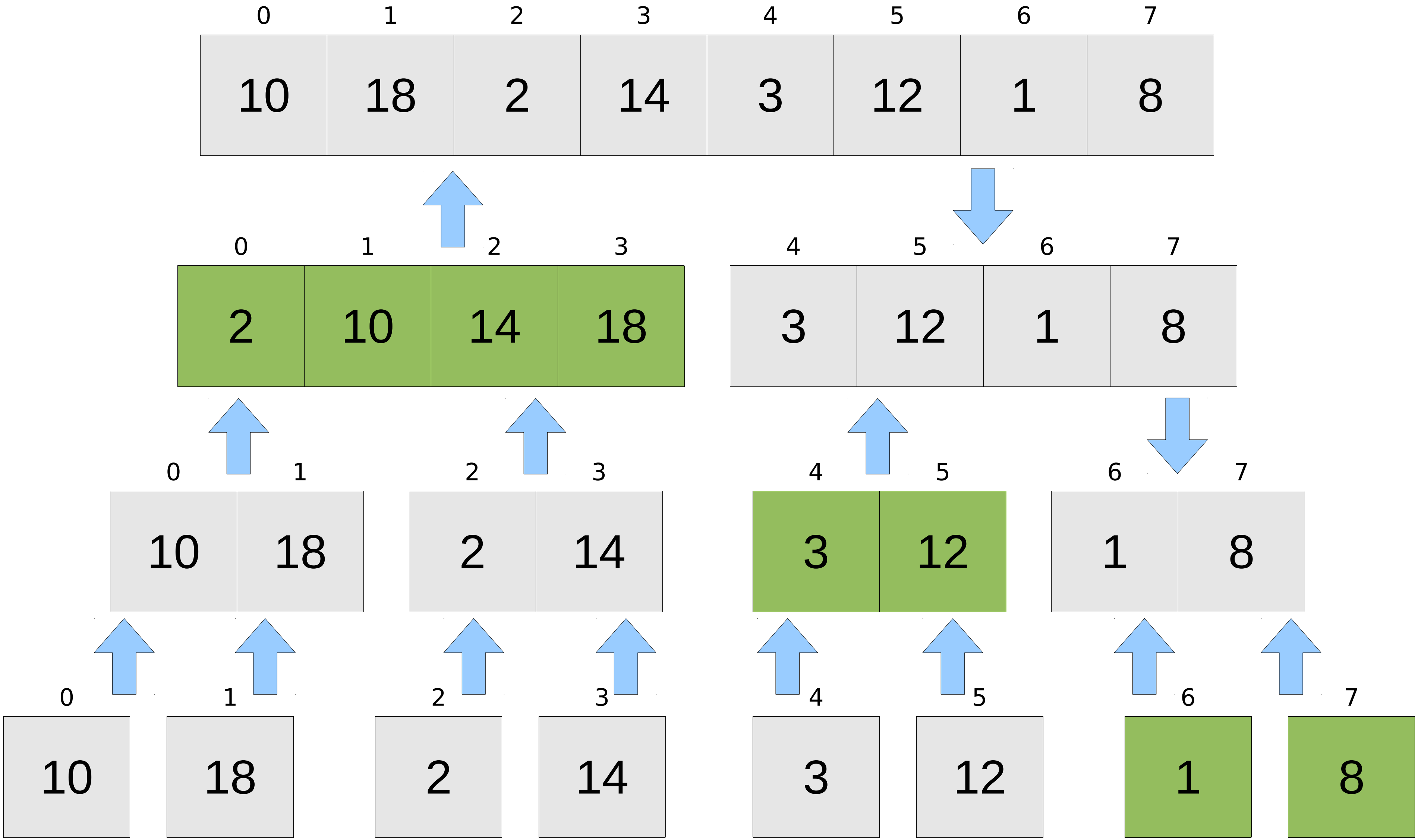
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

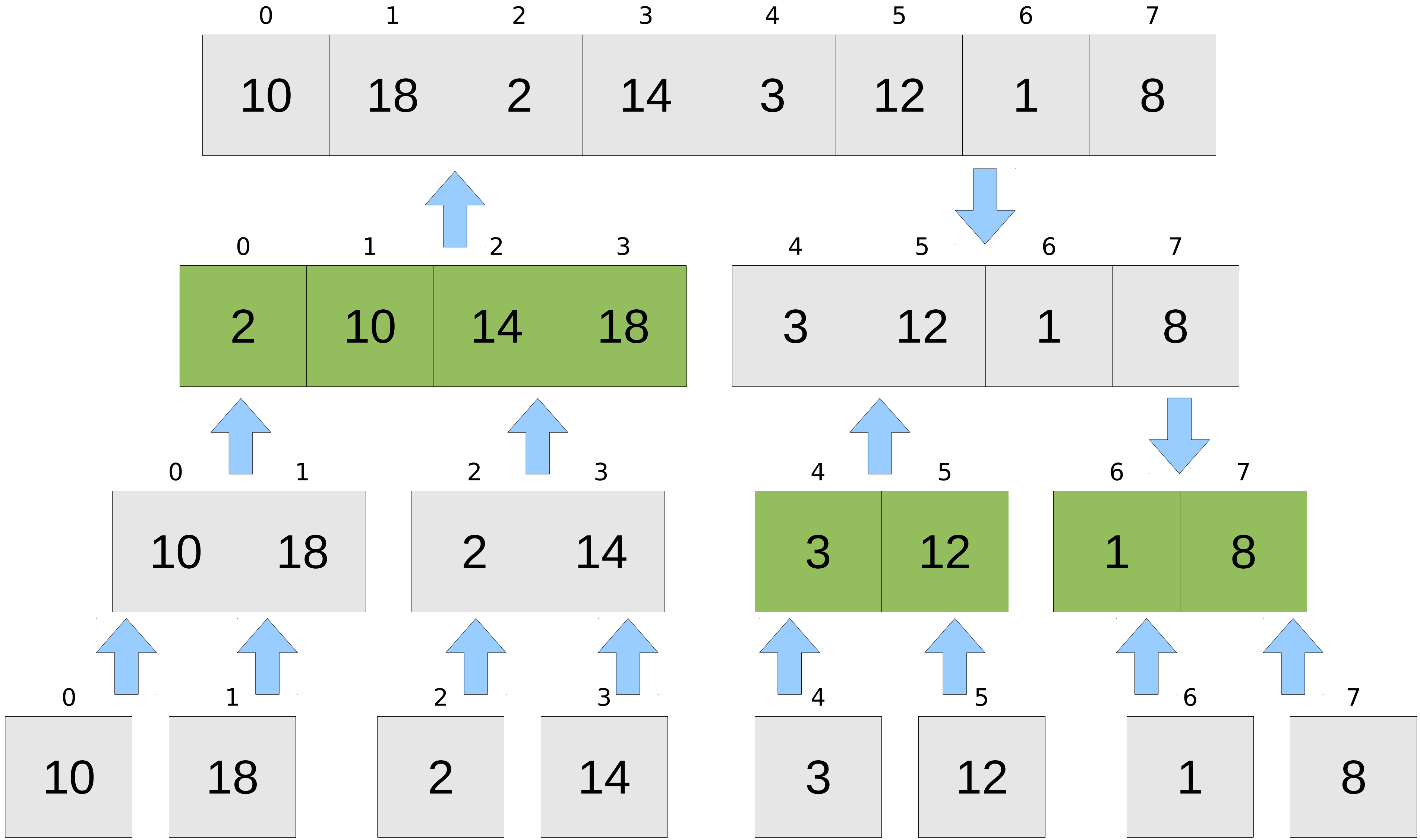
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

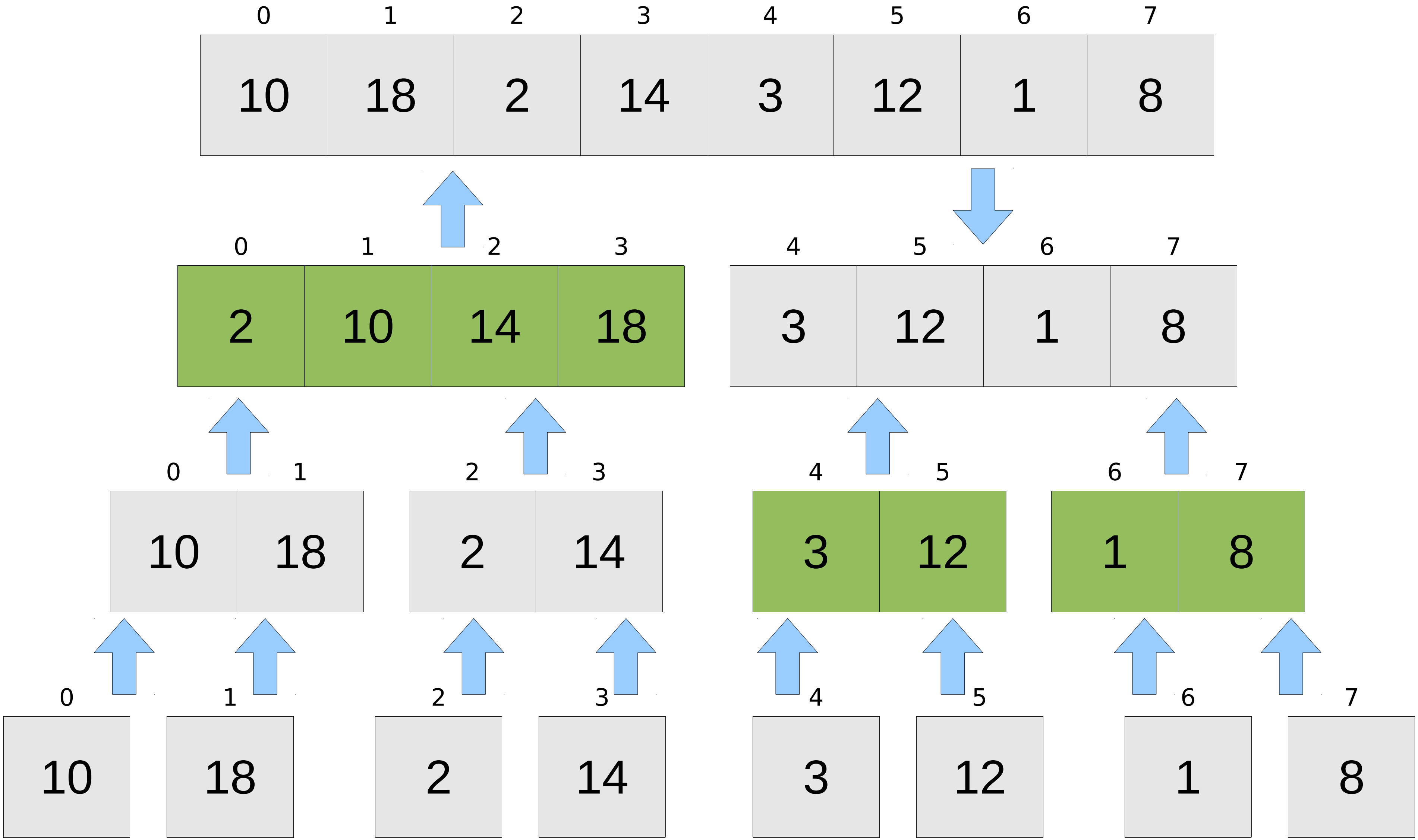
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
3	12	1	8

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

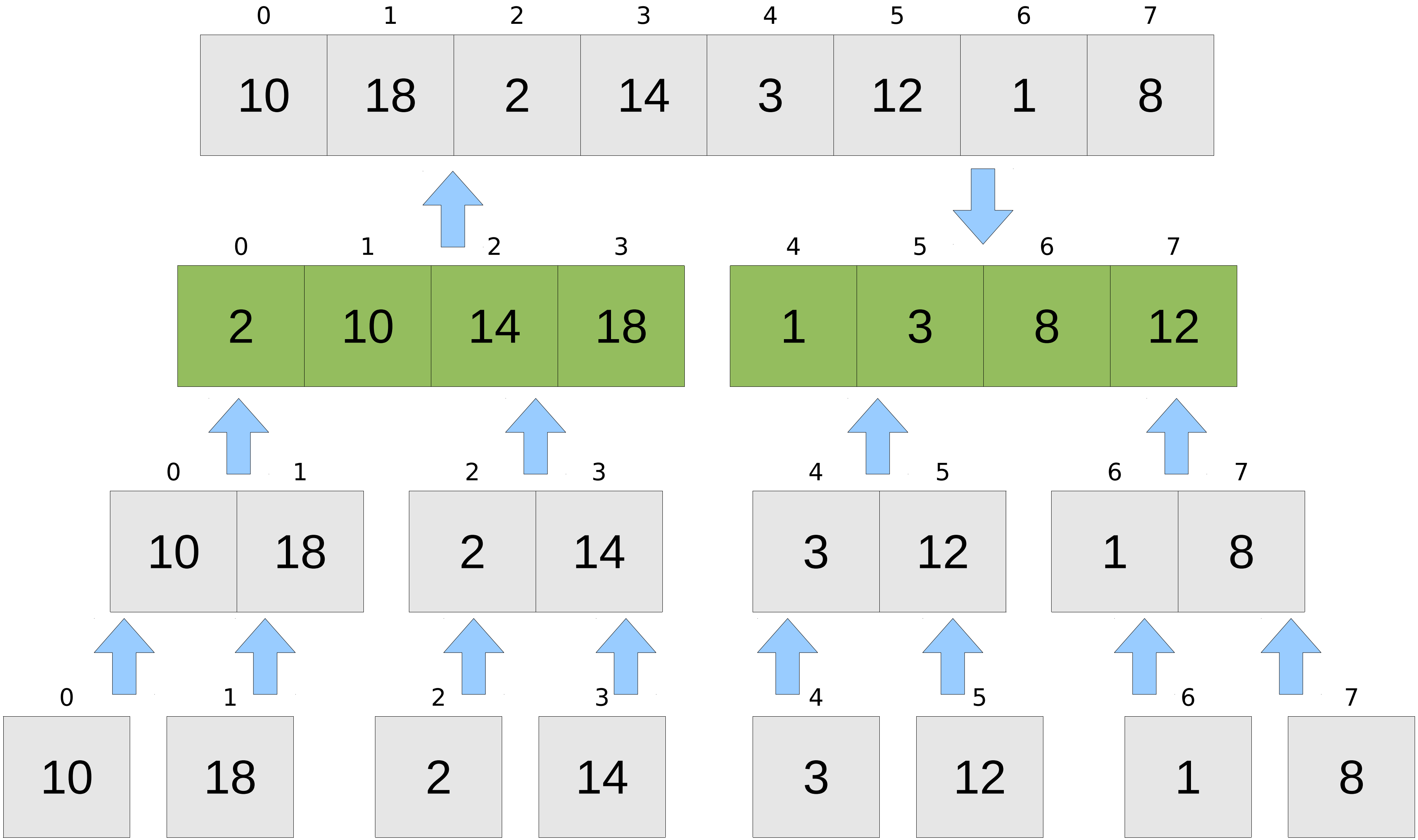
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

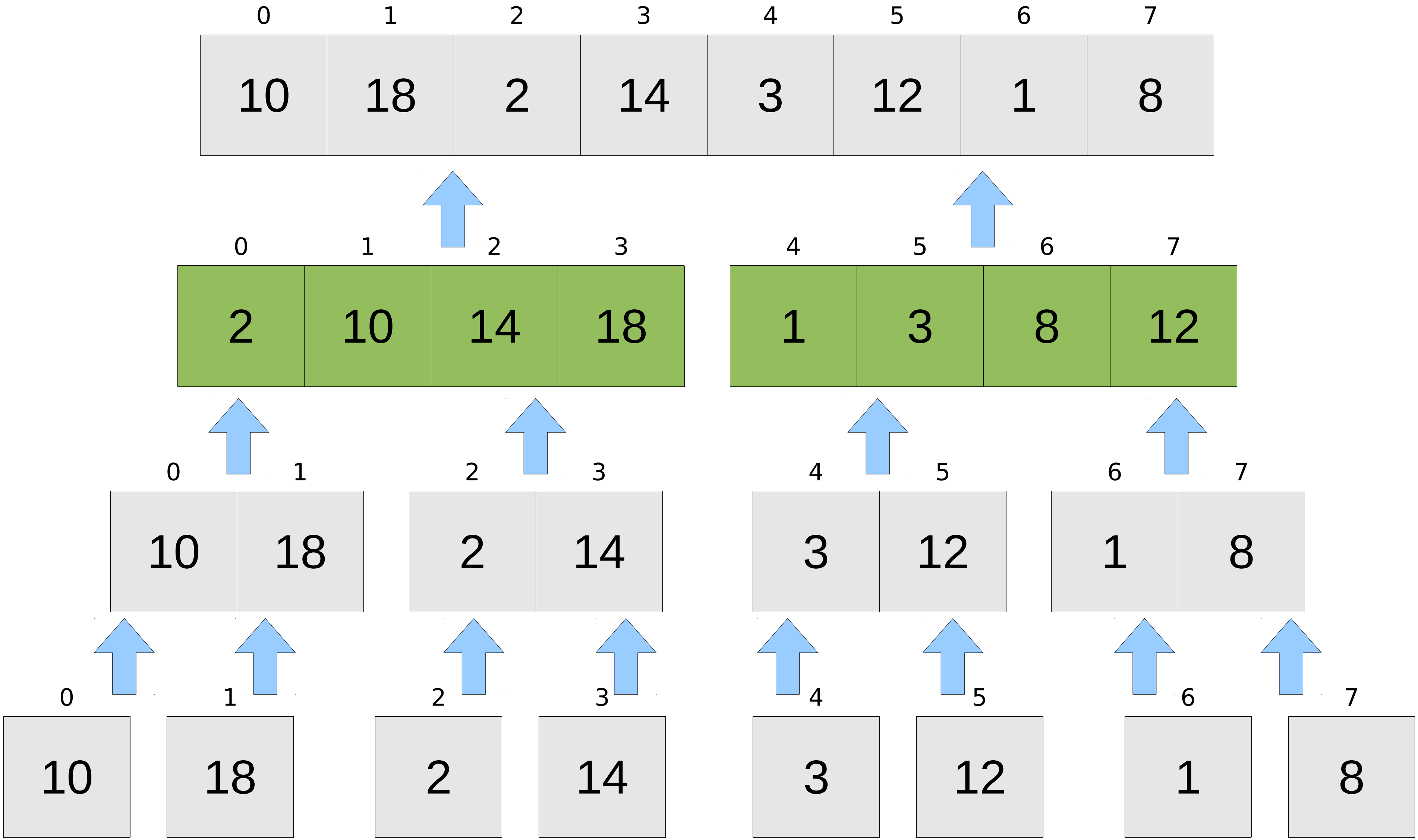
6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1
10	18

2	3
2	14

4	5
3	12

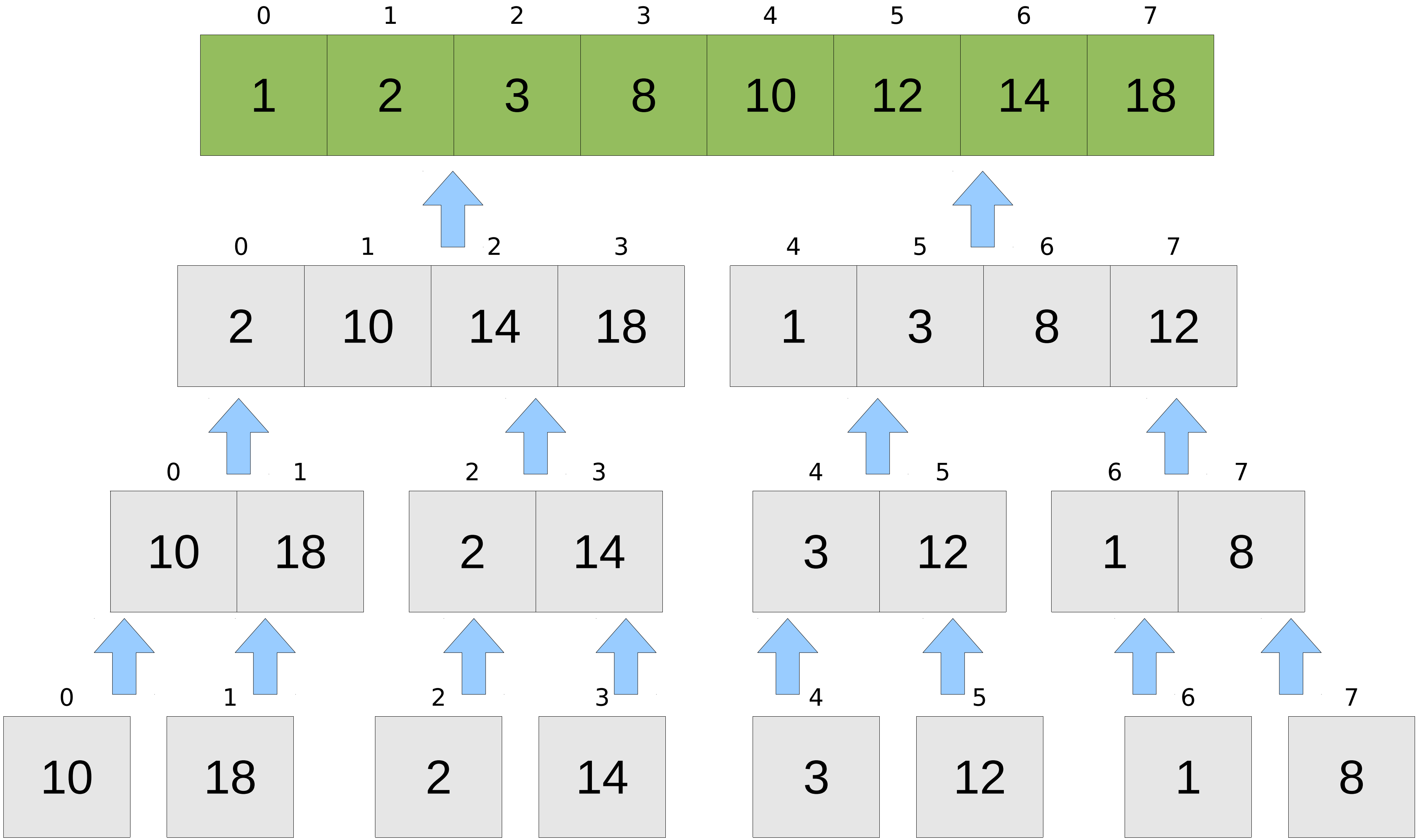
6	7
1	8

0	1	2	3	4	5	6	7
10	18	2	14	3	12	1	8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)



0	1	2	3	4	5	6	7
1	2	3	8	10	12	14	18

0	1	2	3
2	10	14	18

4	5	6	7
1	3	8	12

0	1
10	18

2	3
2	14

4	5
3	12

6	7
1	8

0
10

1
18

2
2

3
14

4
3

5
12

6
1

7
8

We’ve reached our **base cases**. An array with one element is **sorted**.

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
1	2	3	8	10	12	14	18

*** TADA! ***

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
1	2	3	8	10	12	14	18

1. What’s the **worst-case** Big-Oh runtime?

2. What’s the **best-case** Big-Oh runtime?

3. On an exam: trace the results of **each recursive sub-call** to Merge Sort.

Merge Sort

(break down the array, then merge the pieces back together)

0	1	2	3	4	5	6	7
1	2	3	8	10	12	14	18

Let’s code it up!