Bubble Sort

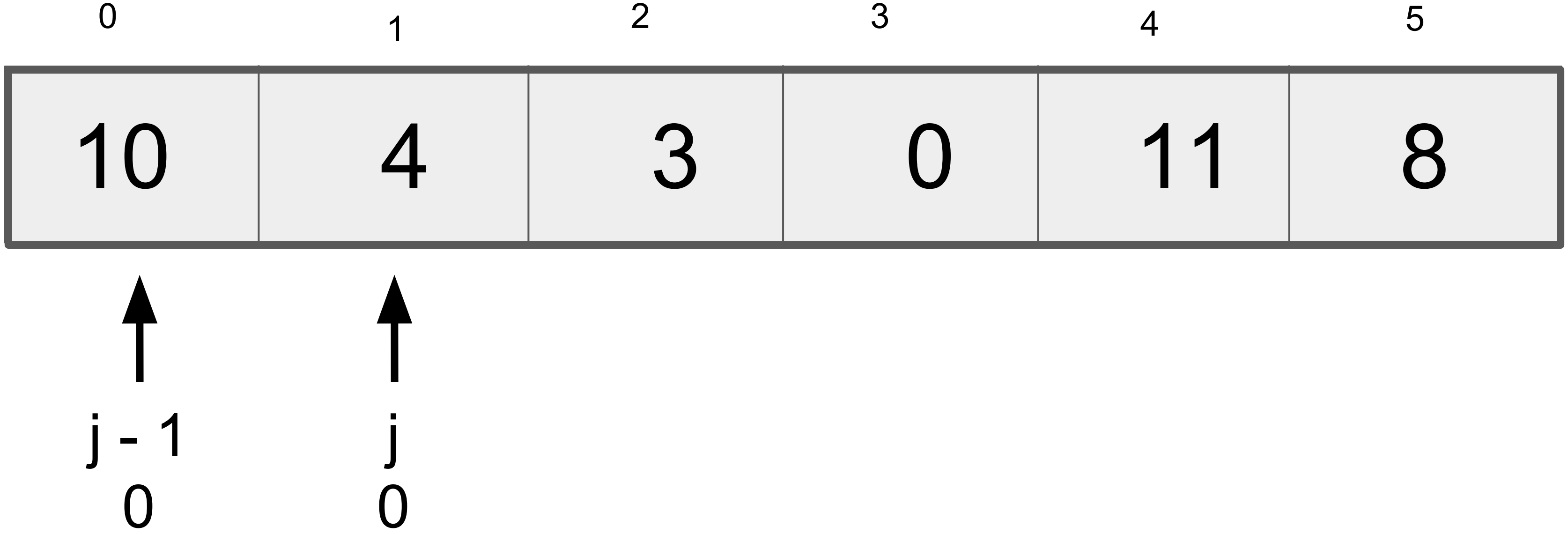
Bubble Sort

0	1	2	3	4	5
10	4	3	0	11	8

What do we do first?

Bubble Sort

len = 6



0	1	2	3	4	5
10	4	3	0	11	8

j - 1
0

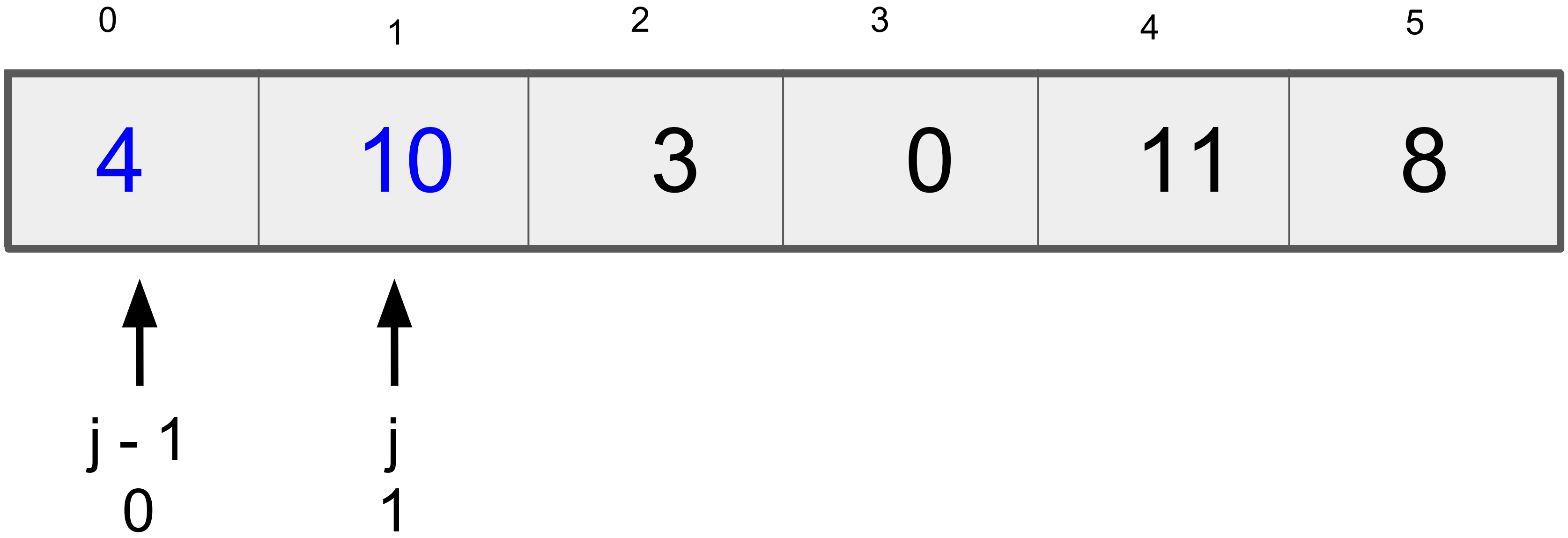
j
0

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	10	3	0	11	8

j - 1
0

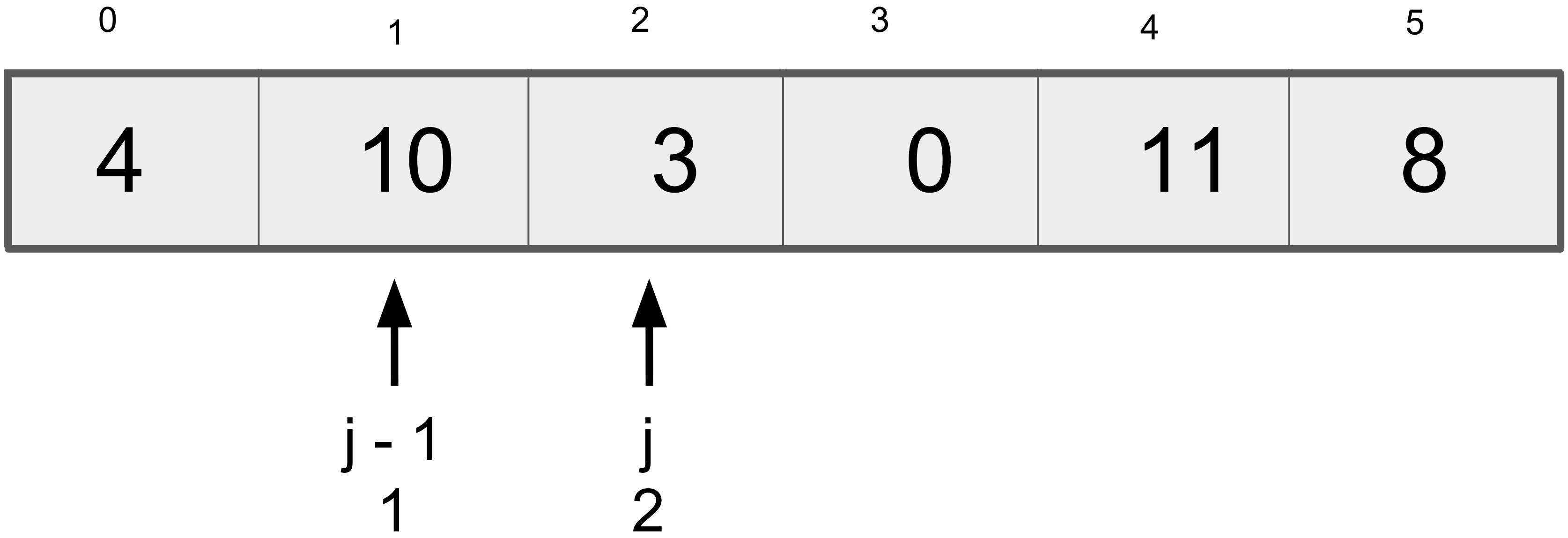
j
1

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



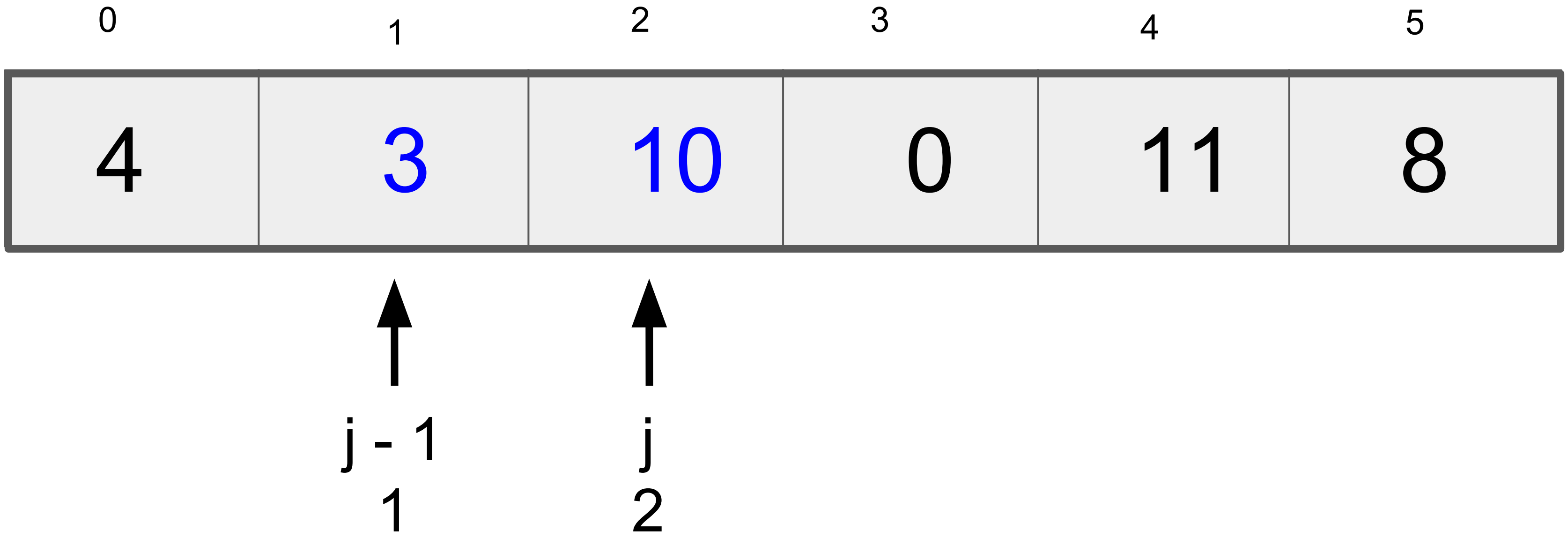
0	1	2	3	4	5
4	10	3	0	11	8

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



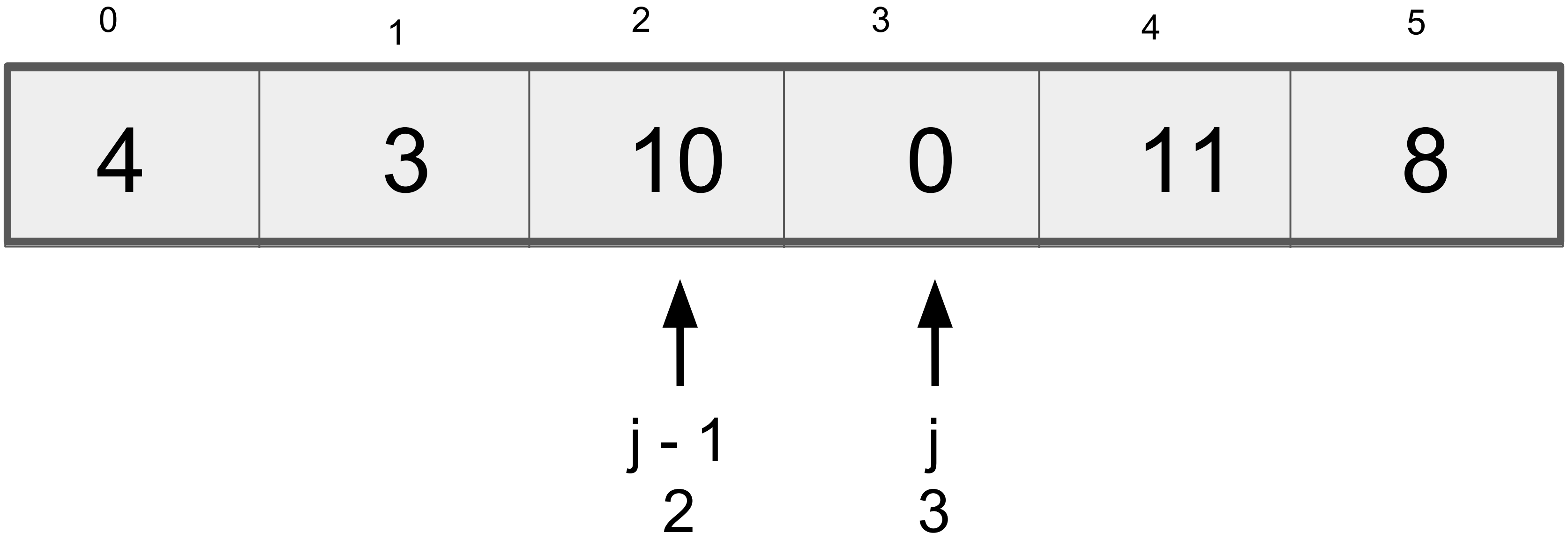
0	1	2	3	4	5
4	3	10	0	11	8

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	3	10	0	11	8

j - 1
2

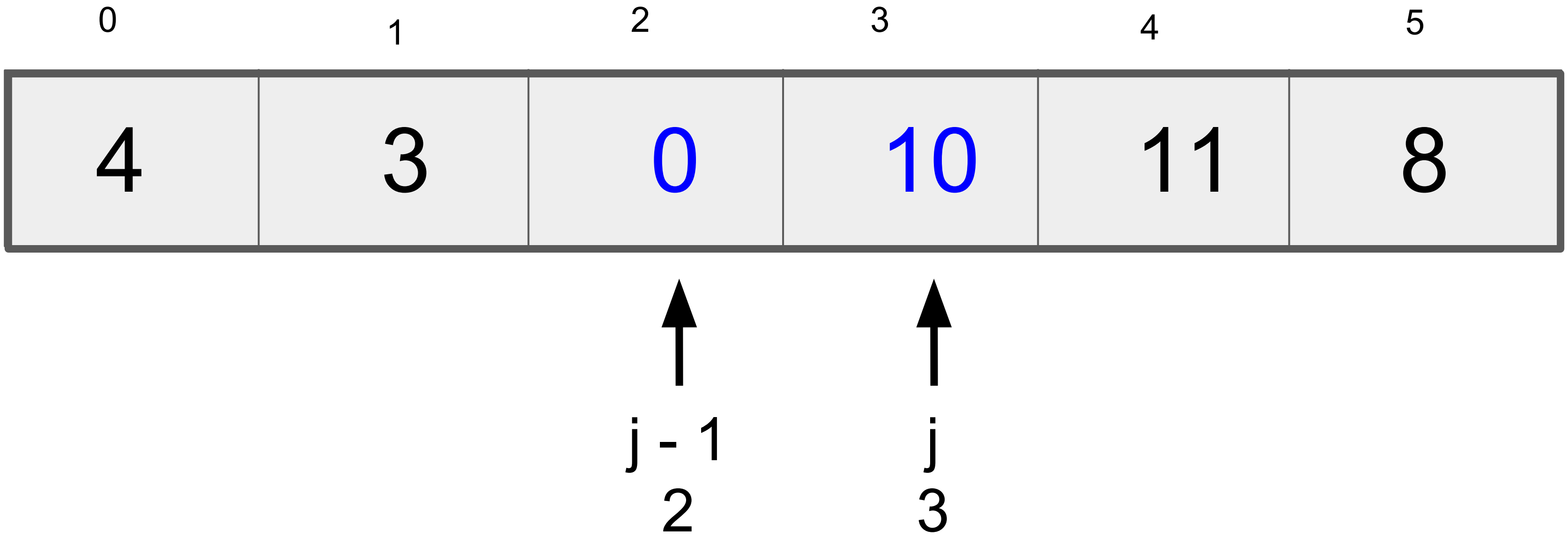
j
3

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	3	0	10	11	8

j - 1
2

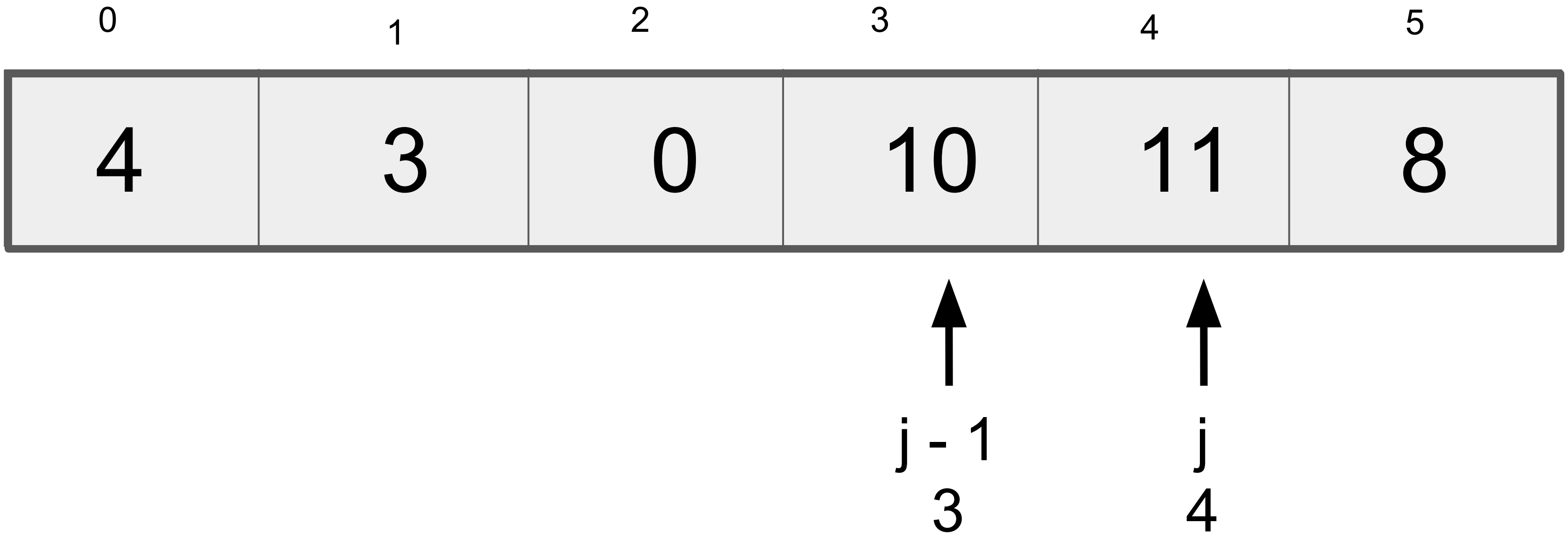
j
3

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	3	0	10	11	8

j - 1
3

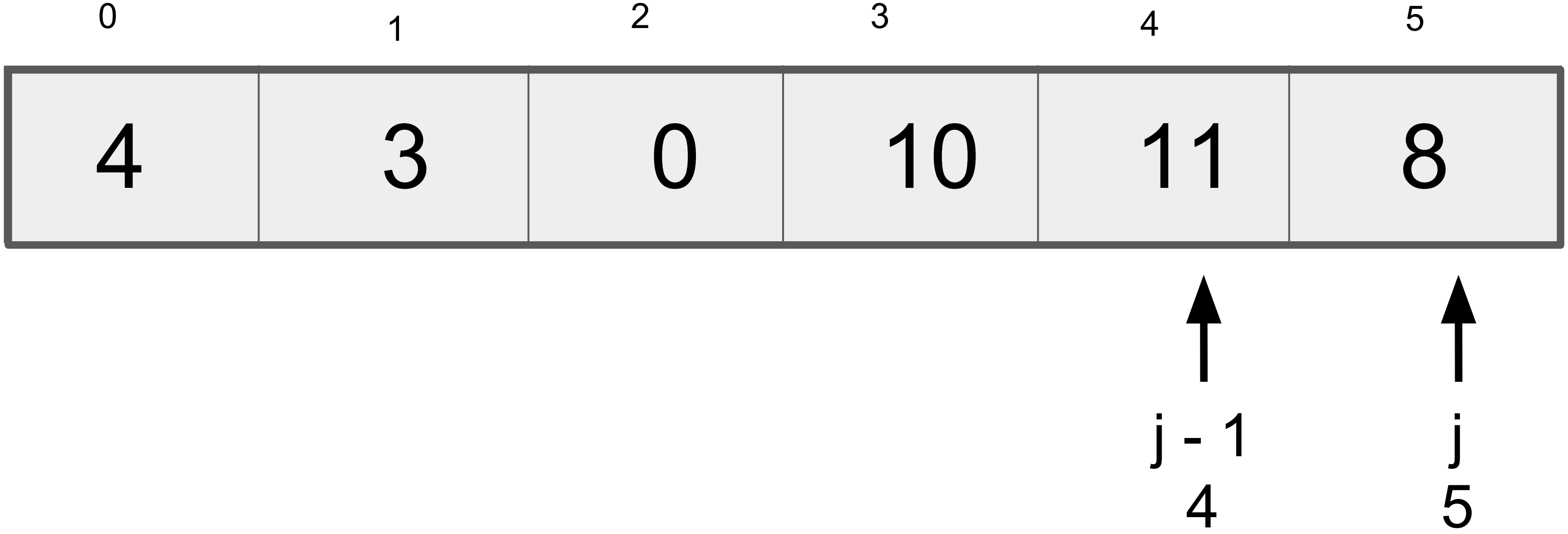
j
4

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	3	0	10	11	8

j - 1
4

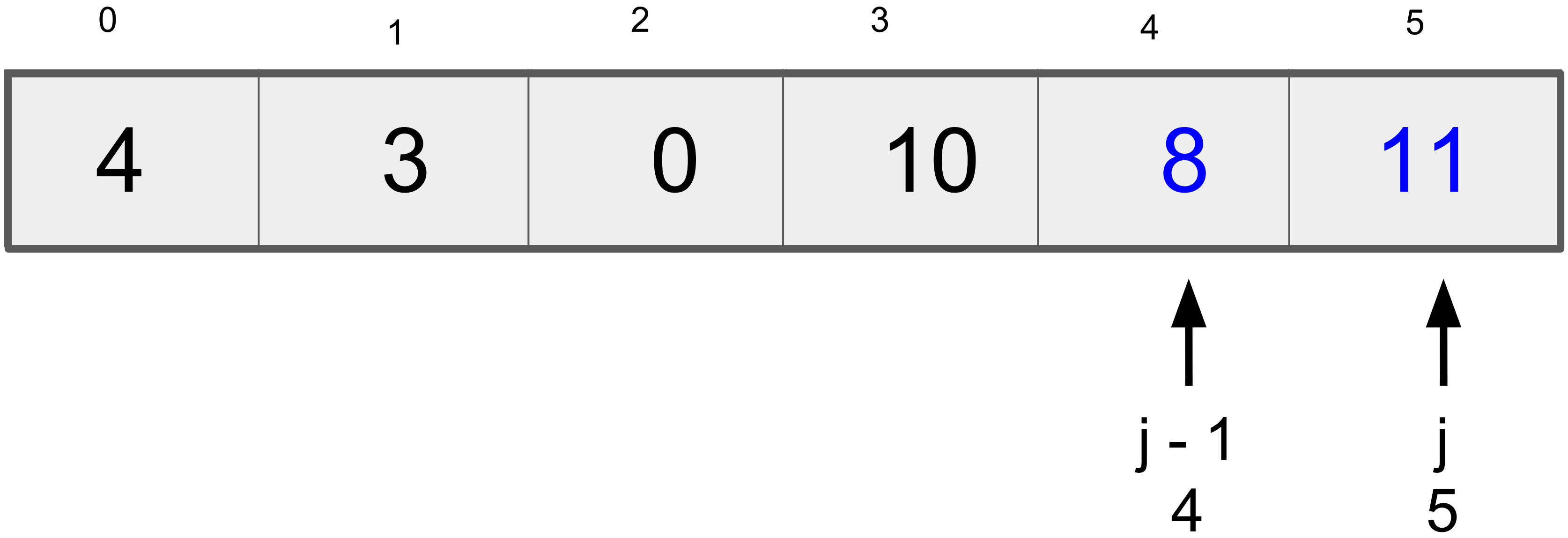
j
5

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 6



0	1	2	3	4	5
4	3	0	10	8	11

j - 1
4

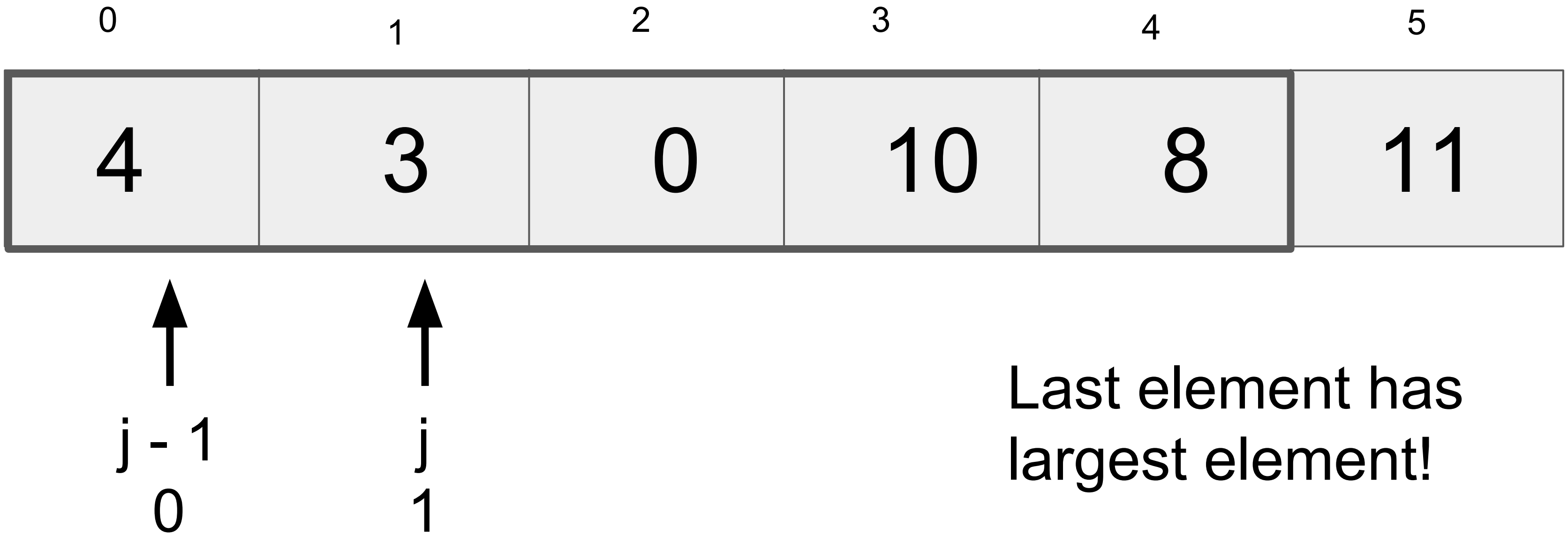
j
5

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



0	1	2	3	4	5
4	3	0	10	8	11

j - 1
0

j
1

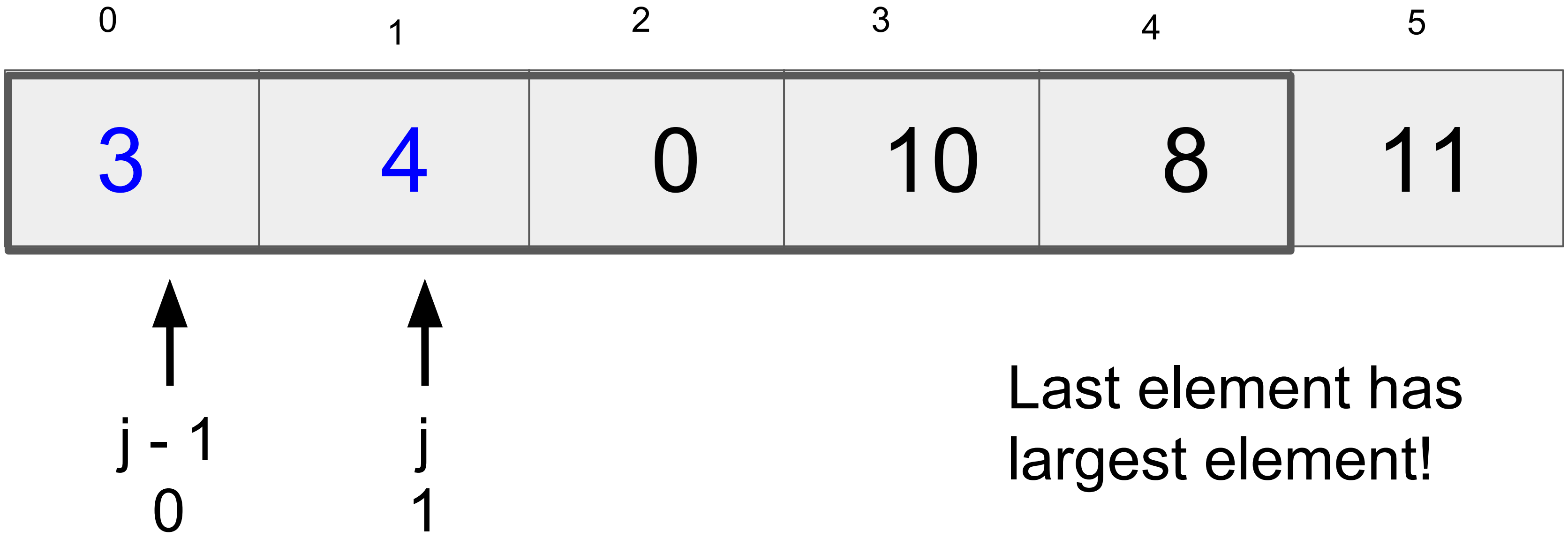
Last element has
largest element!

Reset and compare pairs with shorter list!

What next?

Bubble Sort

len = 5



0	1	2	3	4	5
3	4	0	10	8	11

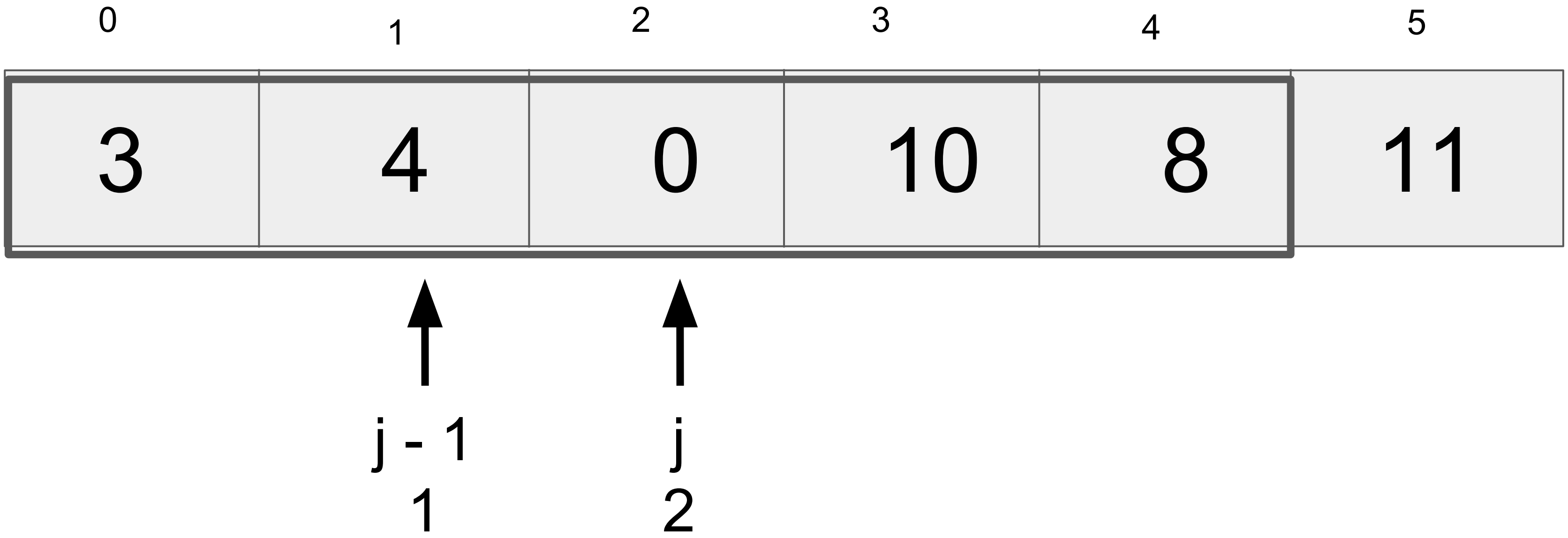
Last element has largest element!

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



0	1	2	3	4	5
3	4	0	10	8	11

j - 1
1

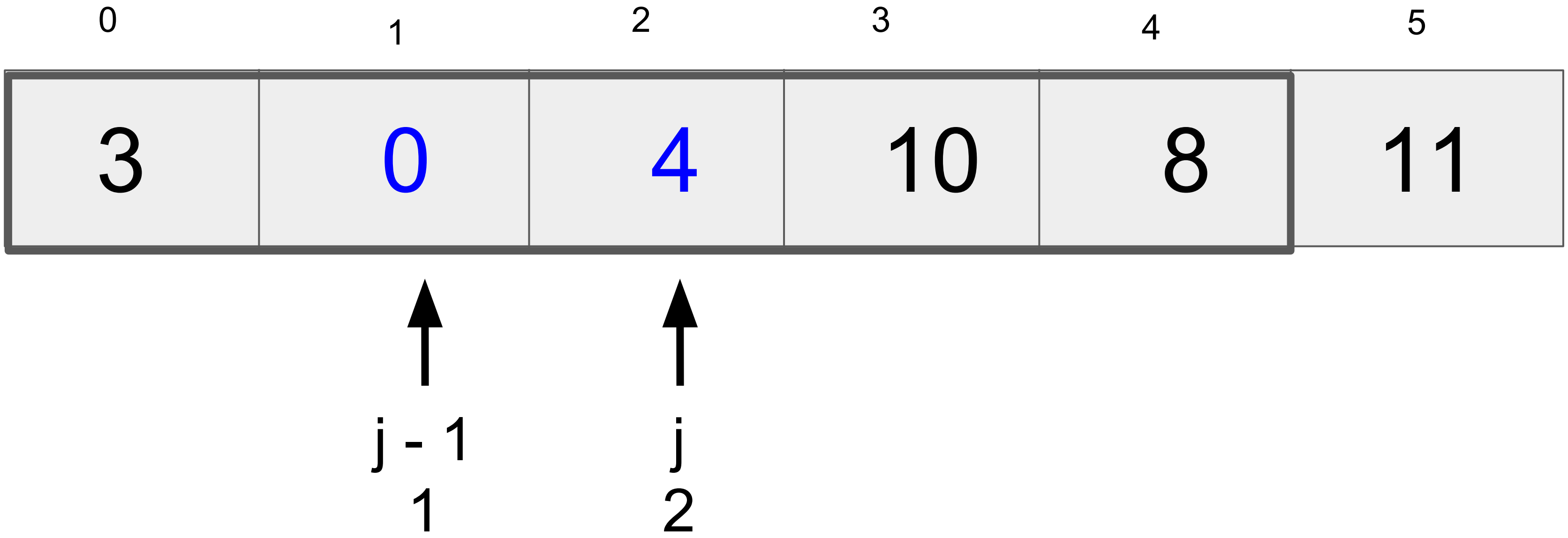
j
2

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



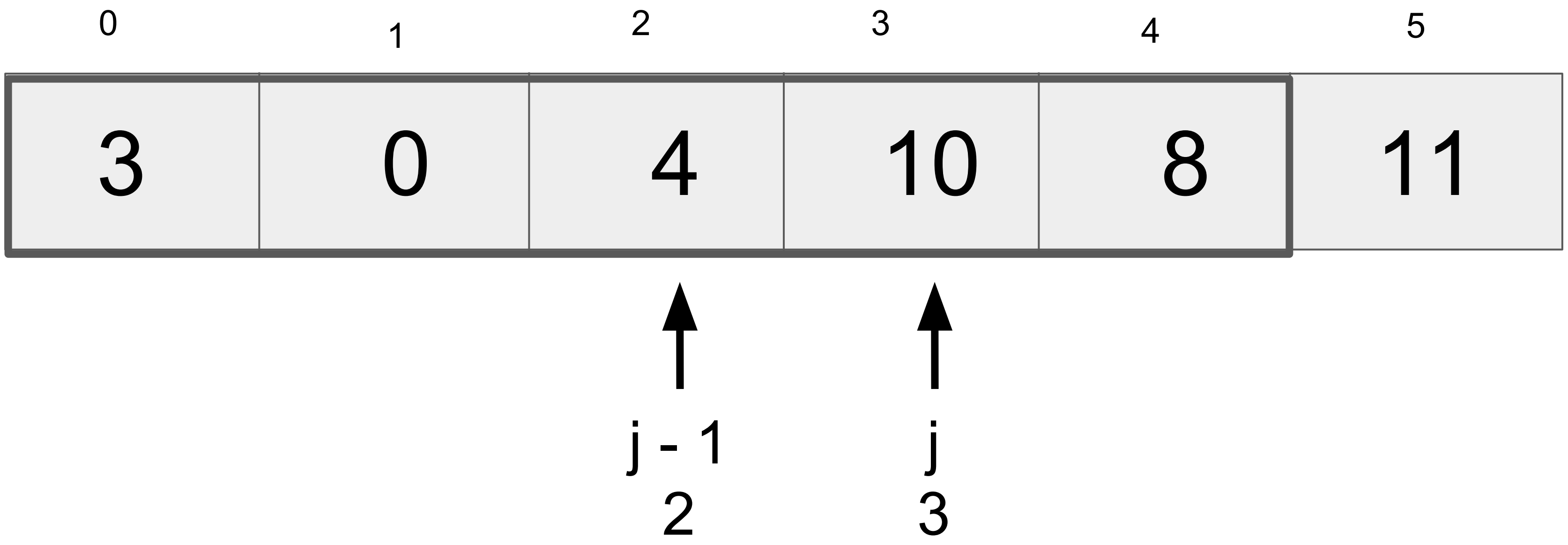
0	1	2	3	4	5
3	0	4	10	8	11

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



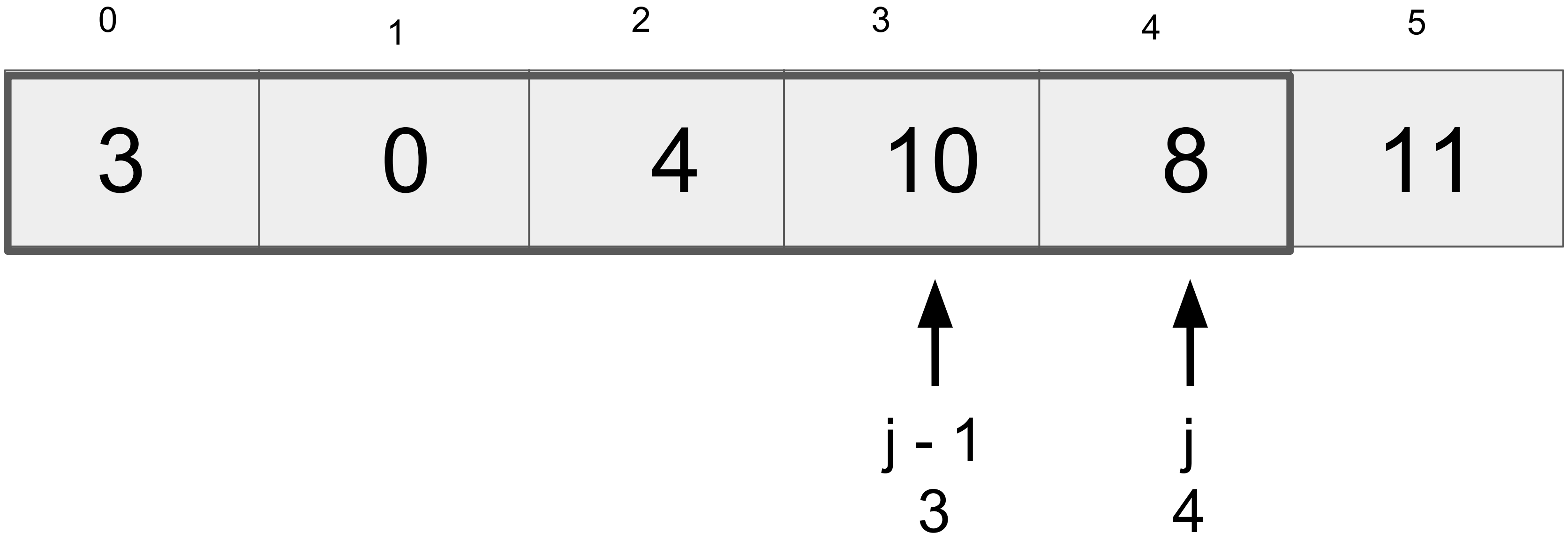
0	1	2	3	4	5
3	0	4	10	8	11

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



0	1	2	3	4	5
3	0	4	10	8	11

j - 1
3

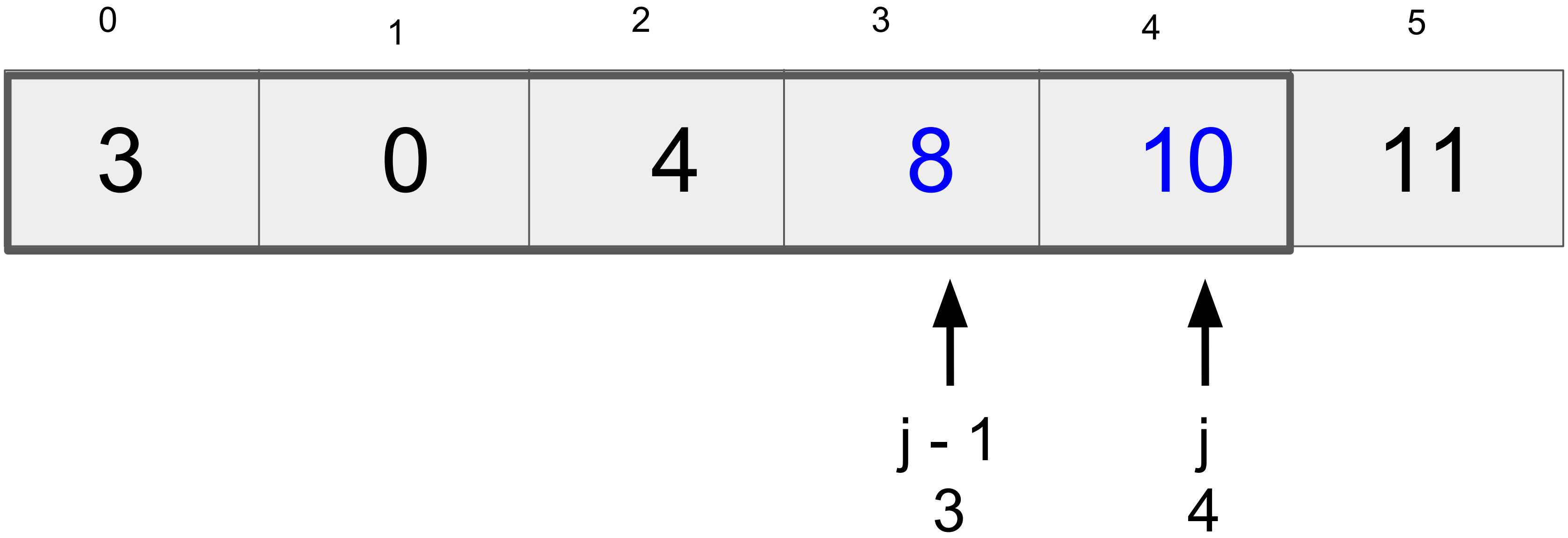
j
4

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 5



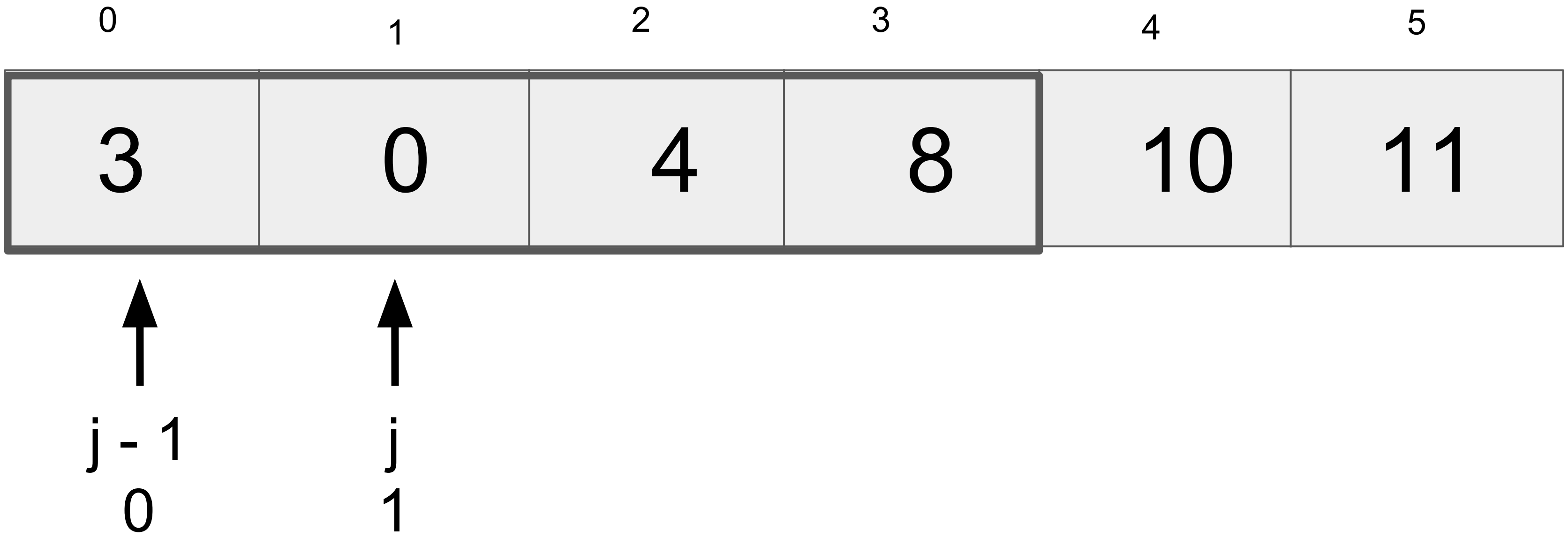
0	1	2	3	4	5
3	0	4	8	10	11

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 4



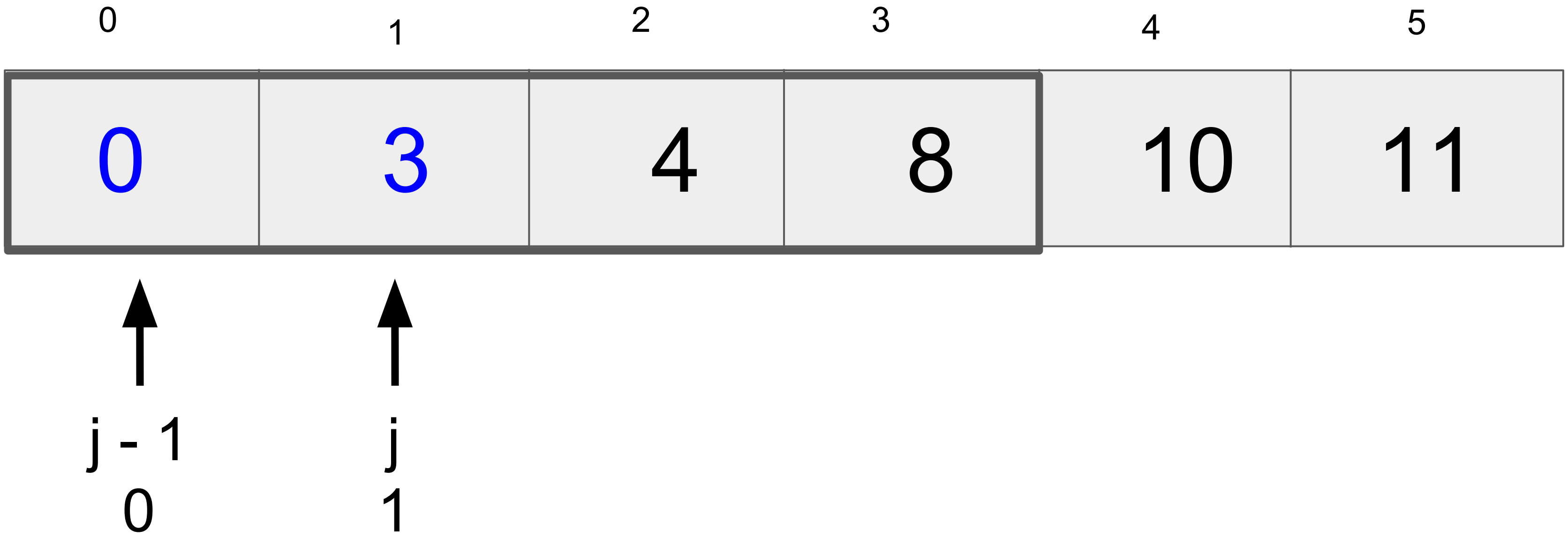
0	1	2	3	4	5
3	0	4	8	10	11

Reset and check pairs with shorter list

What next?

Bubble Sort

len = 4



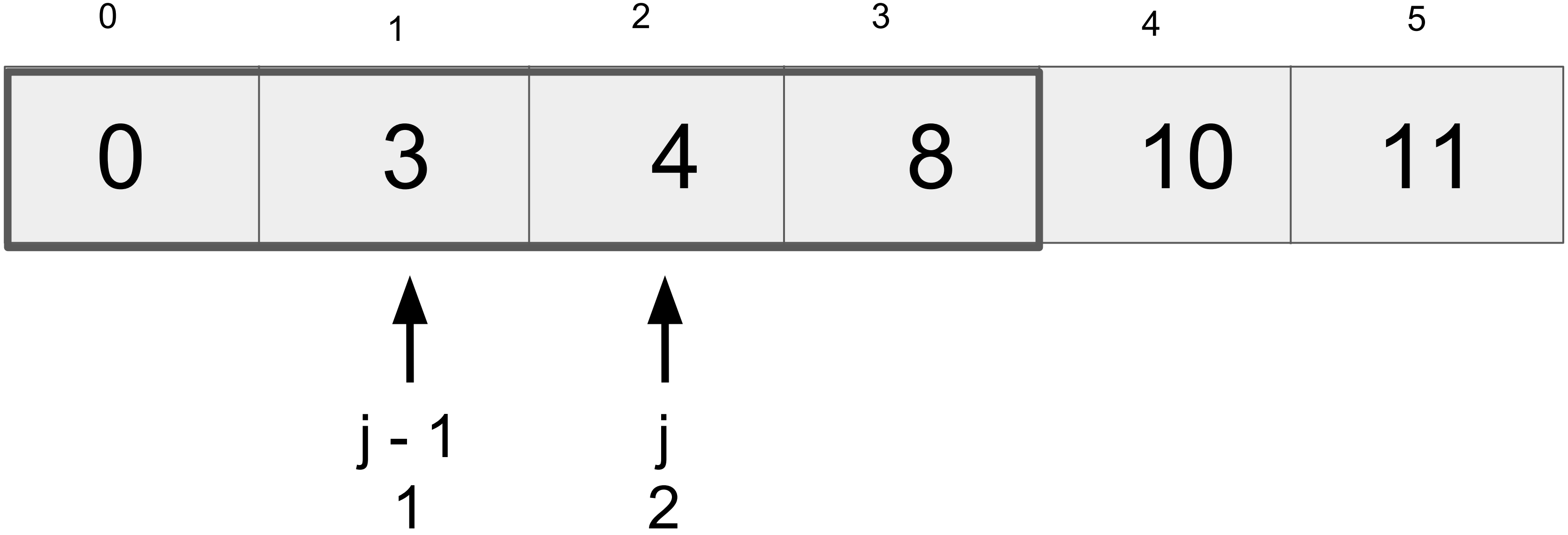
0	1	2	3	4	5
0	3	4	8	10	11

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 4



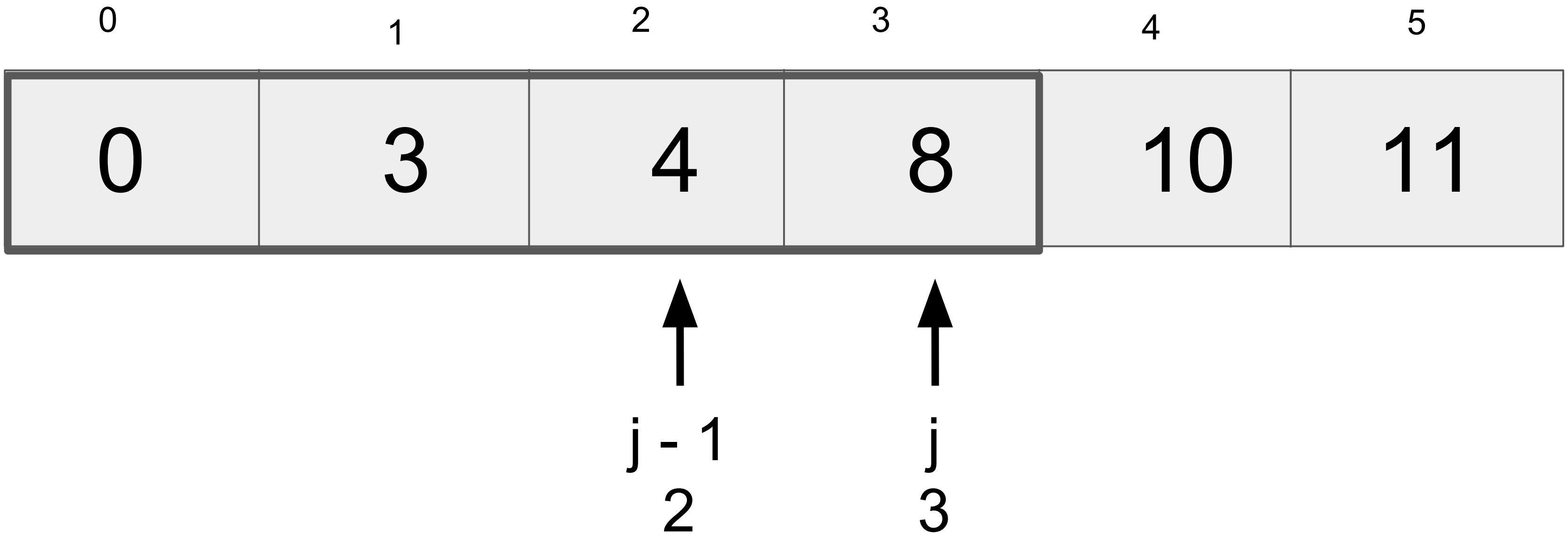
0	1	2	3	4	5
0	3	4	8	10	11

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 4



0	1	2	3	4	5
0	3	4	8	10	11

j - 1
2

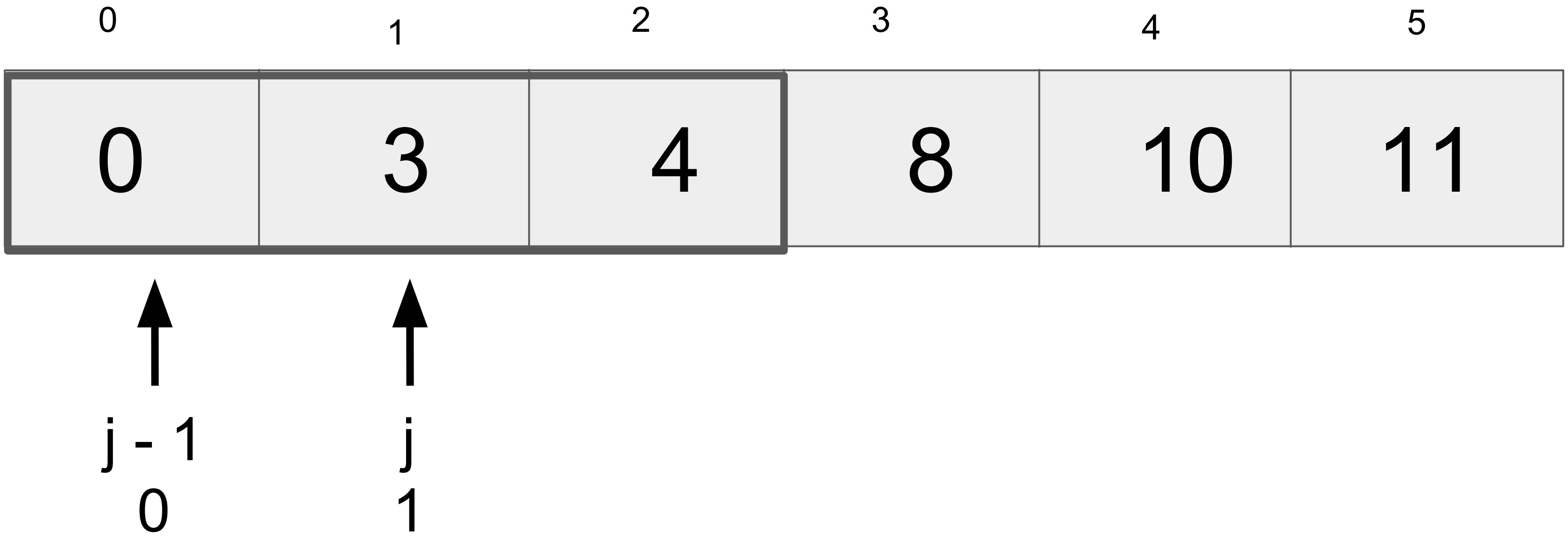
j
3

Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 3



0	1	2	3	4	5
0	3	4	8	10	11

j - 1
0

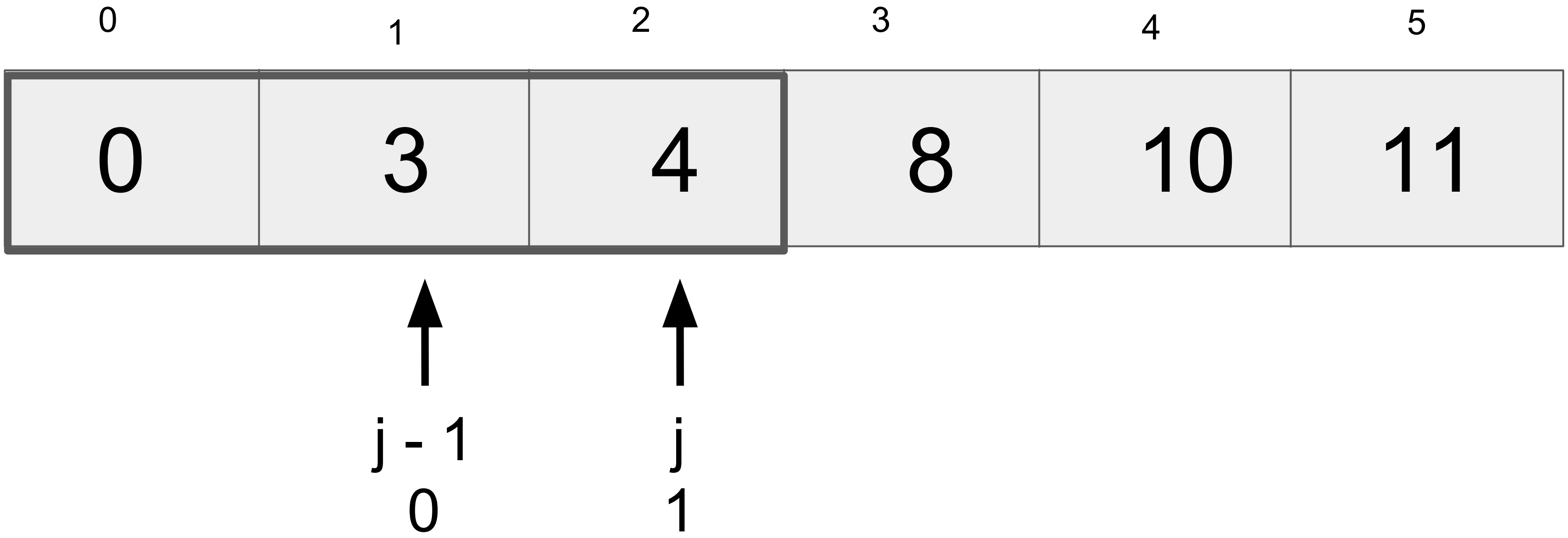
j
1

Reset; Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 3



0	1	2	3	4	5
0	3	4	8	10	11

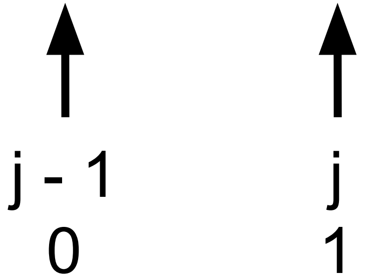
Reset; Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

Bubble Sort

len = 2

0	1	2	3	4	5
0	3	4	8	10	11



Reset; Compare j-1 and j; Swap if L[j-1] > L[j]

What next?

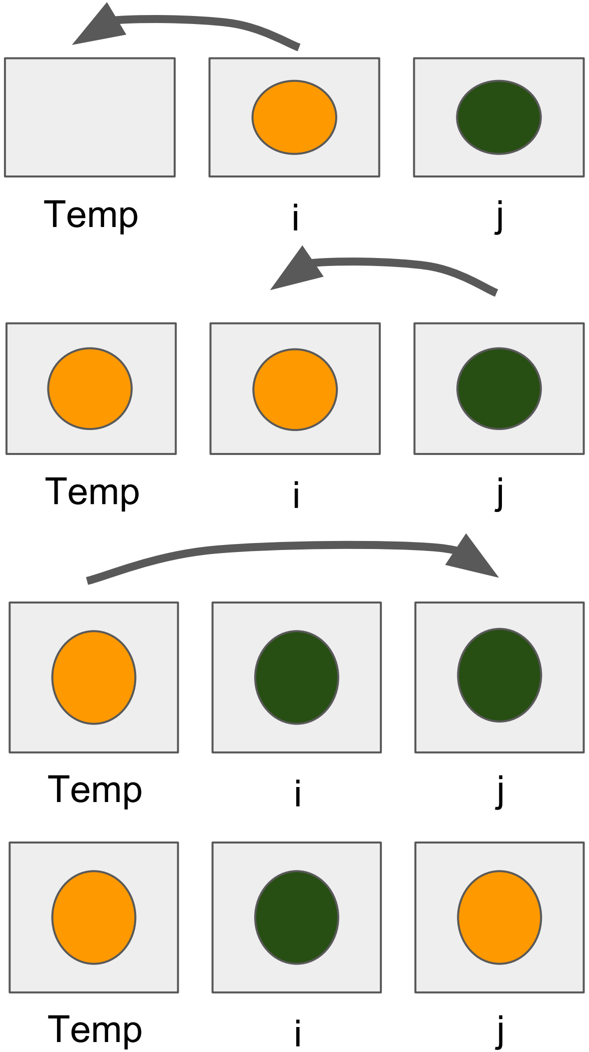
Bubble Sort

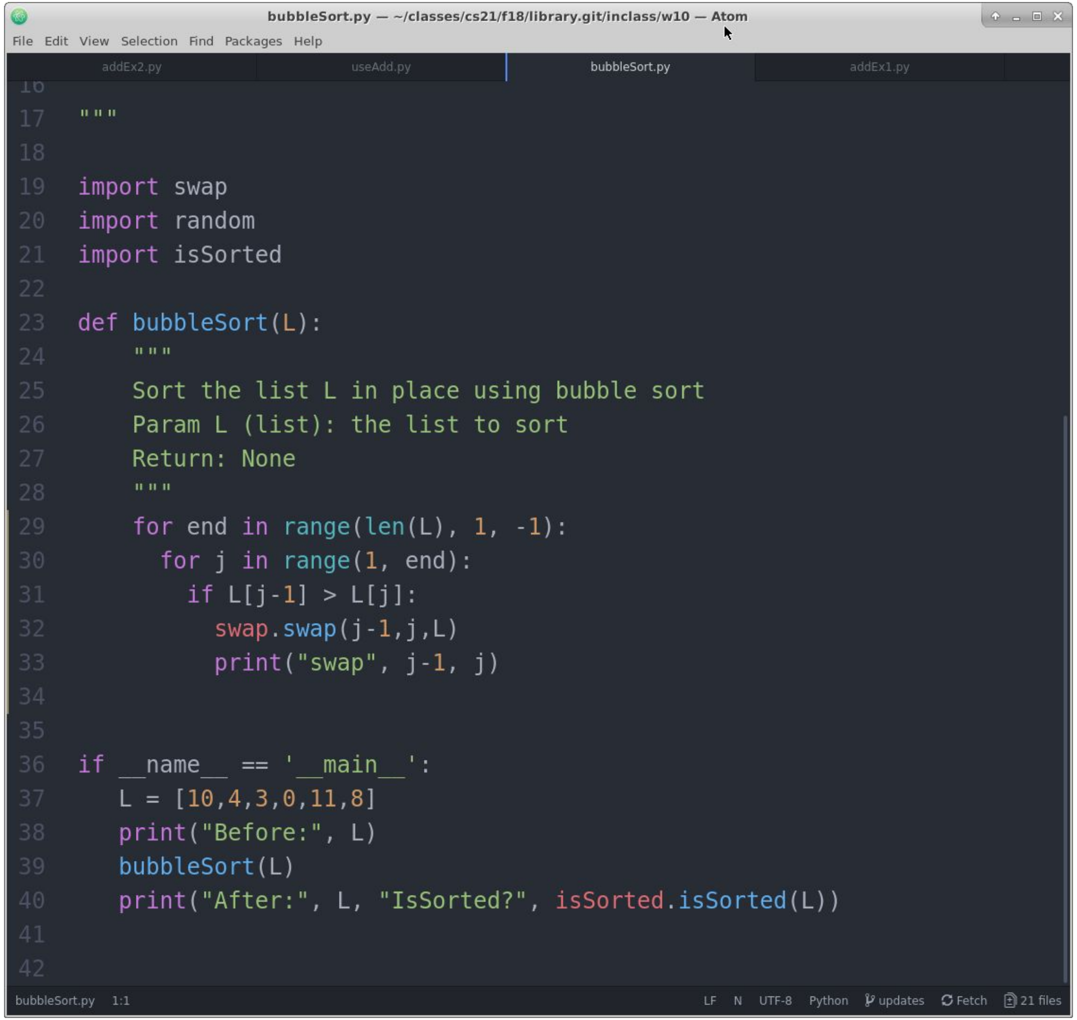
Idea: bubble highest values to the end of the list; Check a shorter sublist each time

```
bubbleSort(L):
    for len in range(len(L), 1, -1):
        for j in range(1, len): # bubble
            if L[j-1] > L[j]:
                swap(j-1, j, L)
```

Bubble sort

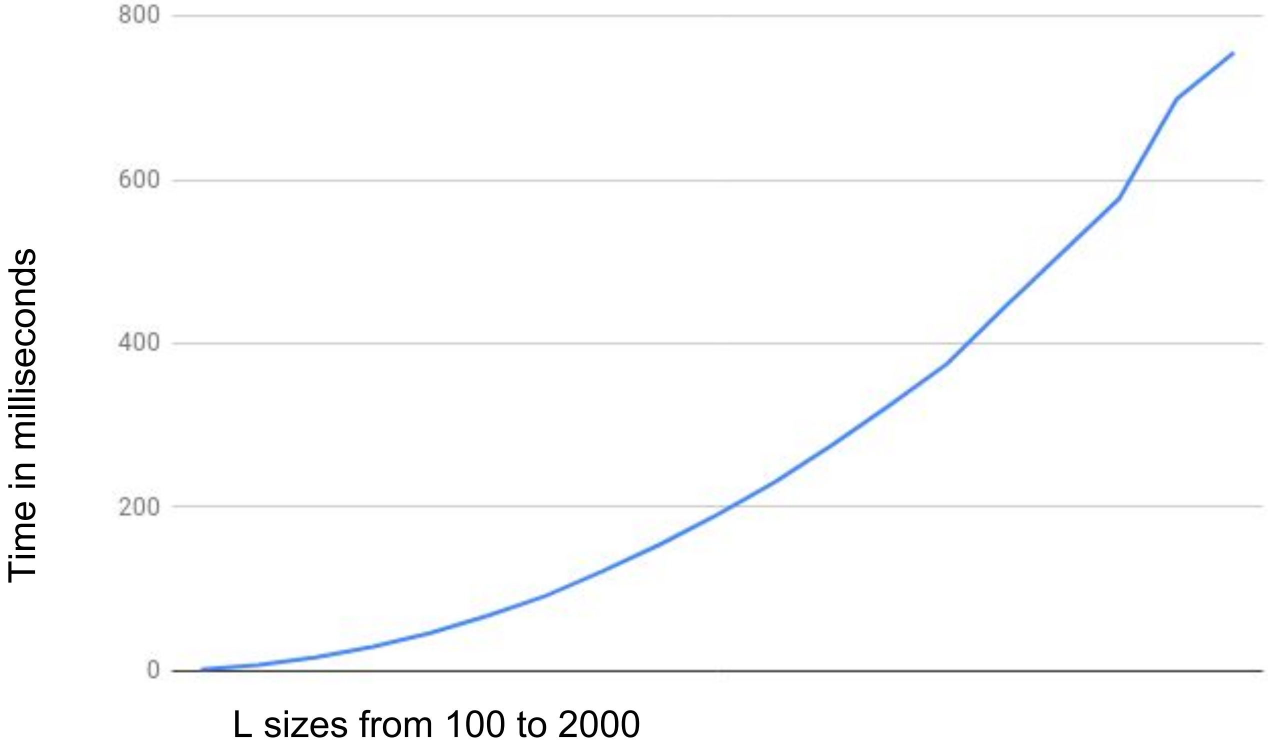
```
swap(i, j, L):
    temp = L[i] # step 1
    L[i] = L[j]    # step 2
    L[j] = temp # step 3
```





```python
"""

import swap
import random
import isSorted

def bubbleSort(L):
    """
    Sort the list L in place using bubble sort
    Param L (list): the list to sort
    Return: None
    """
    for end in range(len(L), 1, -1):
      for j in range(1, end):
        if L[j-1] > L[j]:
          swap.swap(j-1,j,L)
          print("swap", j-1, j)


if __name__ == '__main__':
   L = [10,4,3,0,11,8]
   print("Before:", L)
   bubbleSort(L)
   print("After:", L, "IsSorted?", isSorted.isSorted(L))


```

Selection sort and Bubble sort are O(N^2)


Time in milliseconds
L sizes from 100 to 2000