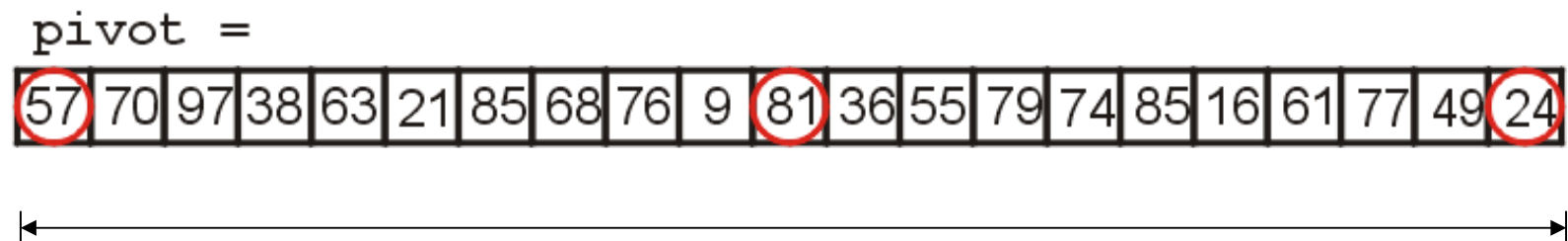


Quick Sort Example

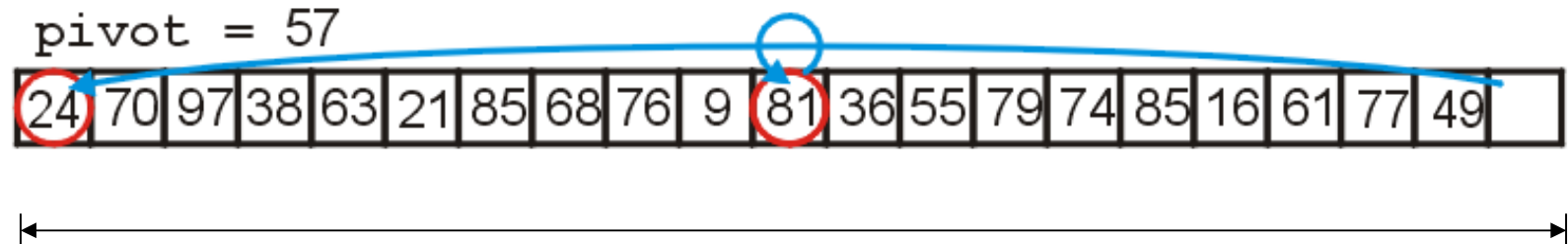
First, we examine the first, middle, and last entries of the full list
The span below will indicate which list we are currently sorting



Quick Sort Example

We select 57 to be our pivot

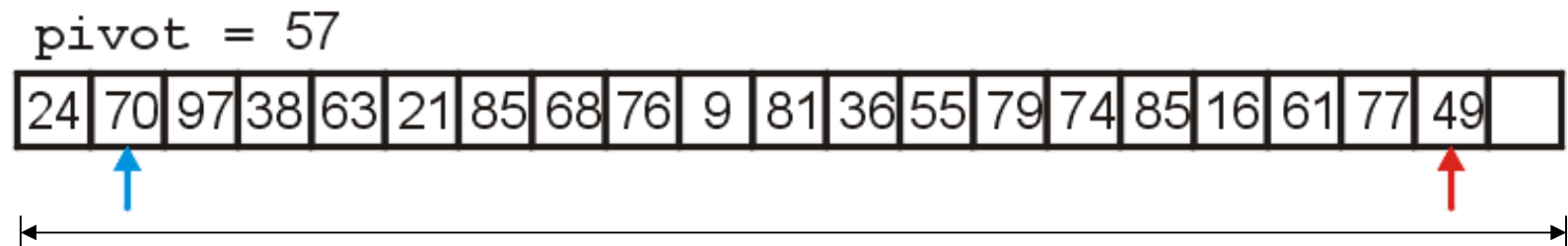
- We move 24 into the first location



Quick Sort Example

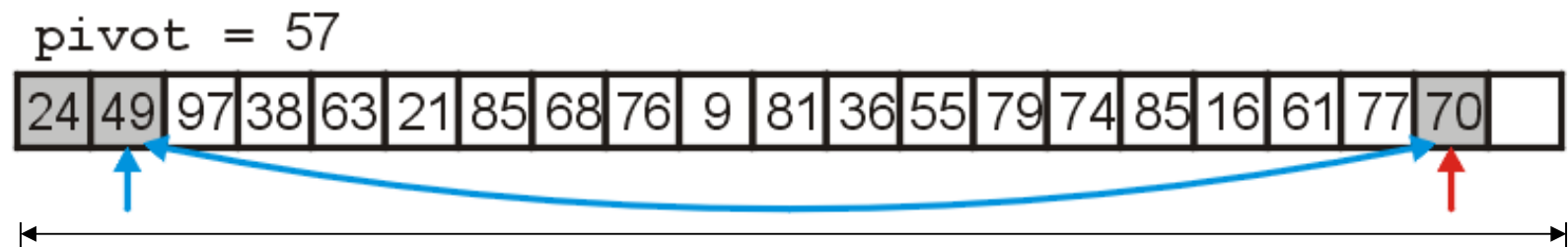
Starting at the 2nd and 2nd-last locations:

- we search forward until we find $70 > 57$
- we search backward until we find $49 < 57$



Quick Sort Example

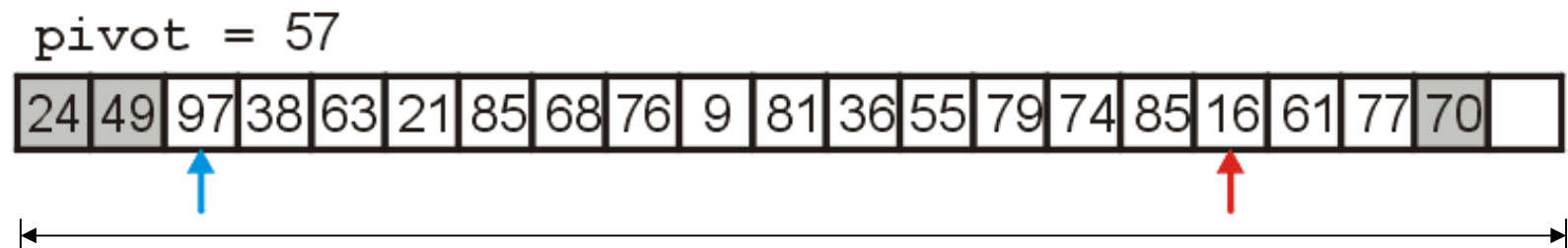
We swap 70 and 49, placing them in order with respect to each other



Quick Sort Example

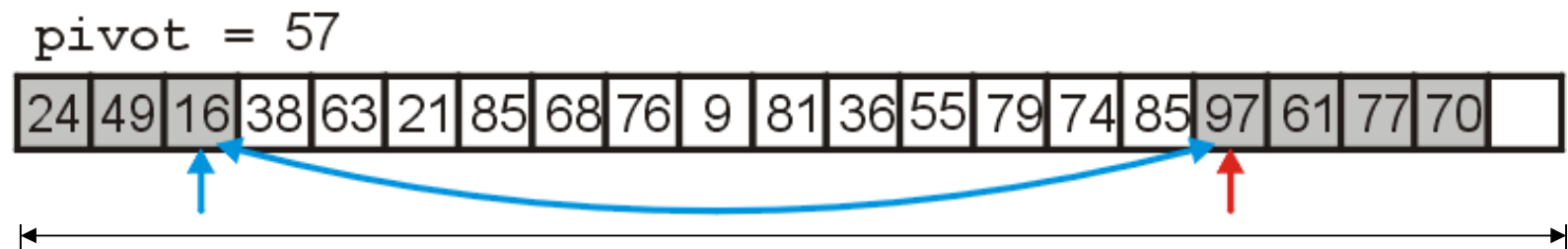
We search forward until we find **97** > 57

We search backward until we find **16** < 57



Quick Sort Example

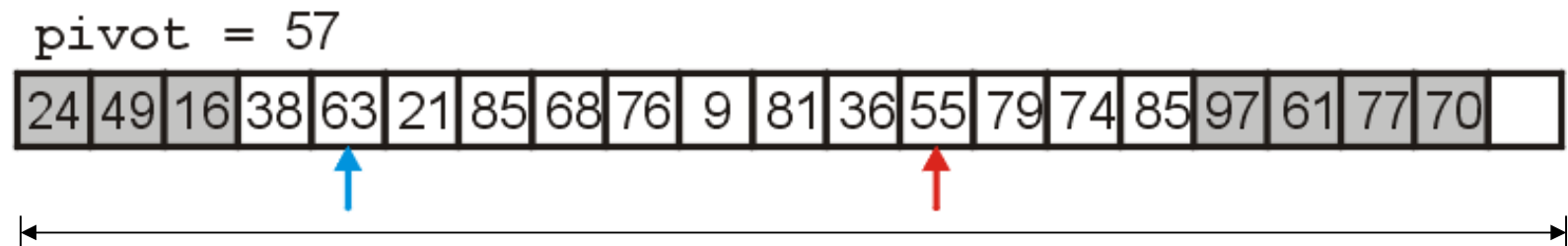
We swap 16 and 97 which are now in order with respect to each other



Quick Sort Example

We search forward until we find **63** > 57

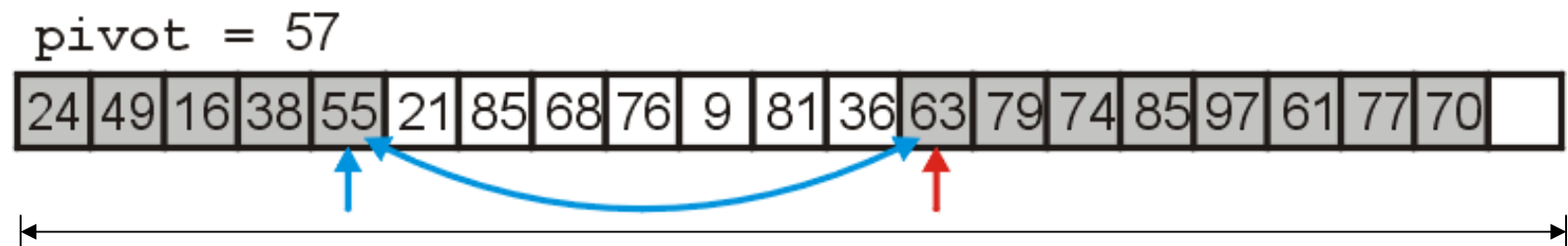
We search backward until we find **55** < 57



Quick Sort

Quick Sort Example

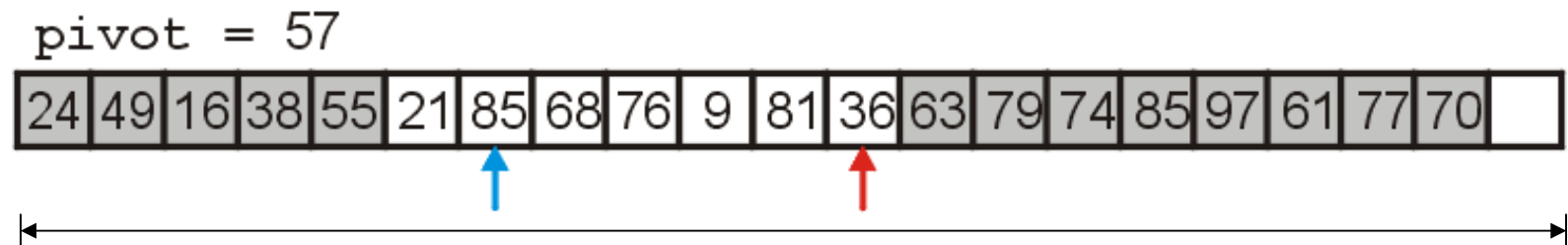
We swap 63 and 55



Quick Sort Example

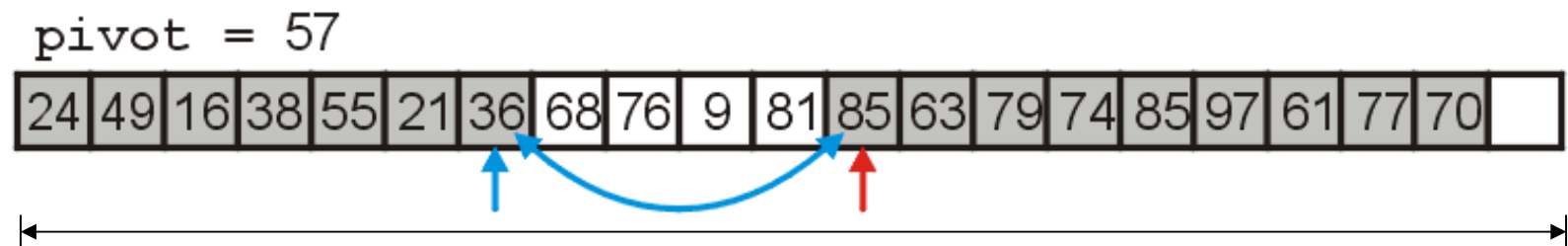
We search forward until we find **85** > 57

We search backward until we find **36** < 57



Quick Sort Example

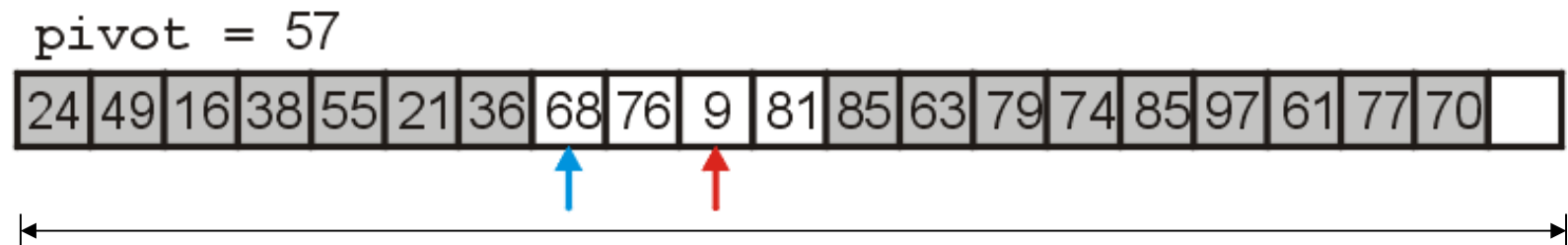
We swap 85 and 36, placing them in order with respect to each other



Quick Sort Example

We search forward until we find $68 > 57$

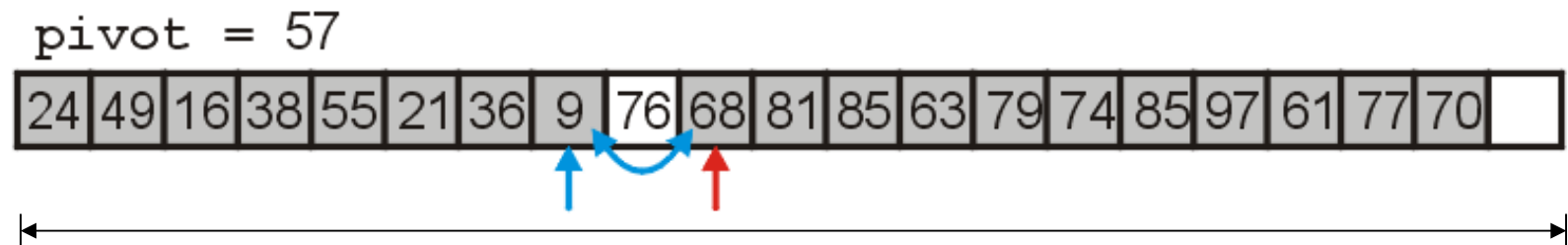
We search backward until we find $9 < 57$



Quick Sort

Quick Sort Example

We swap 68 and 9

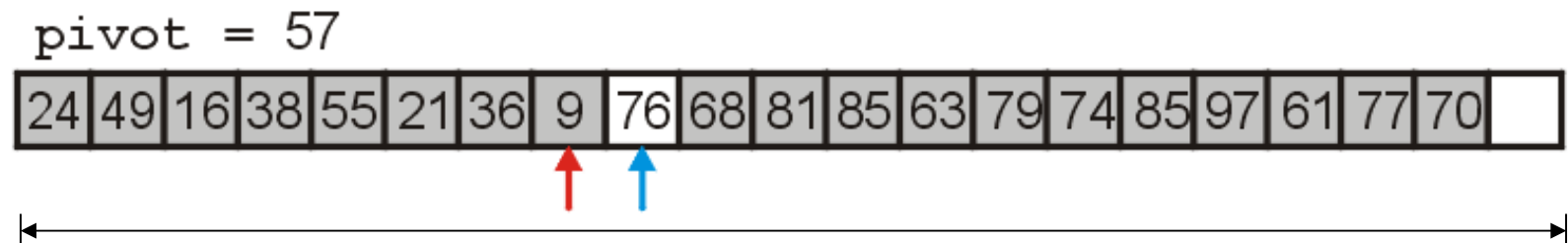


Quick Sort Example

We search forward until we find **76** > 57

We search backward until we find **9** < 57

- The indices are out of order, so we stop

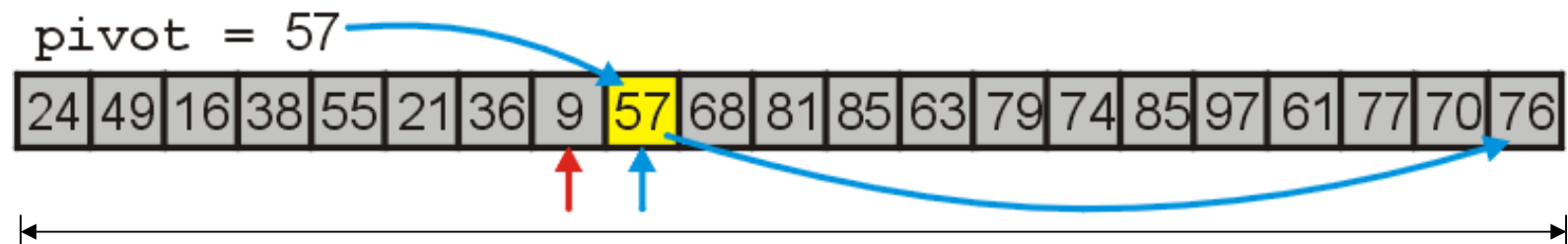


Quick Sort Example

We move the larger indexed item to the vacancy at the end of the array

We fill the empty location with the pivot, 57

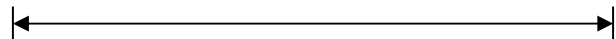
The pivot is now in the correct location



Quick Sort Example

We will now recursively call quick sort on the first half of the list
When we are finished, all entries < 57 will be sorted

24	49	16	38	55	21	36	9	57	68	81	85	63	79	74	85	97	61	77	70	76
----	----	----	----	----	----	----	---	----	----	----	----	----	----	----	----	----	----	----	----	----



Quick Sort Example

We examine the first, middle, and last elements of this sub list

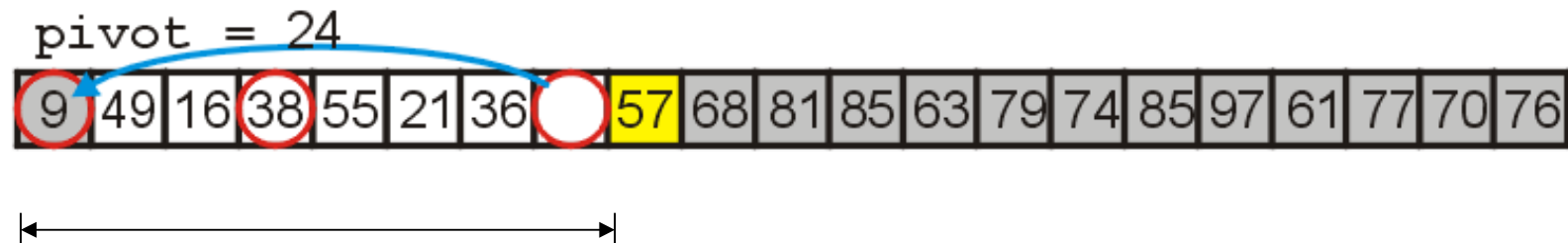
pivot =



Quick Sort Example

We choose 24 to be our pivot

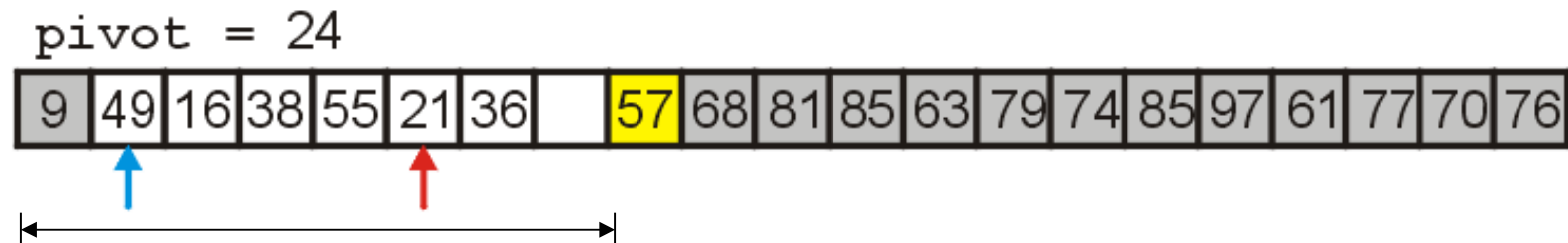
We move 9 into the first location in this sub-list



Quick Sort Example

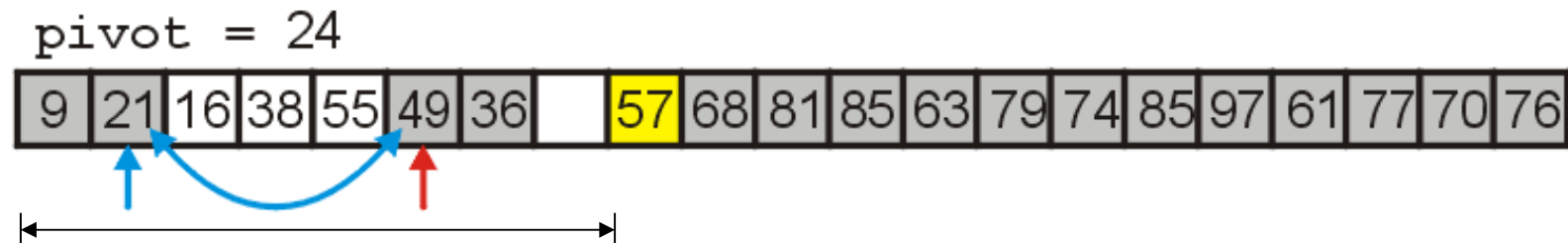
We search forward until we find **49** > 24

We search backward until we find **21** < 24



Quick Sort Example

We swap 49 and 21, placing them in order with respect to each other

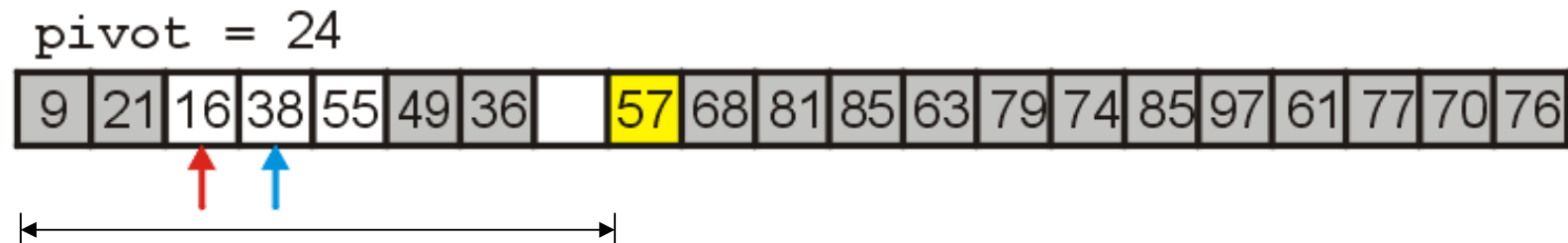


Quick Sort Example

We search forward until we find **38** > 24

We search backward until we find **16** < 24

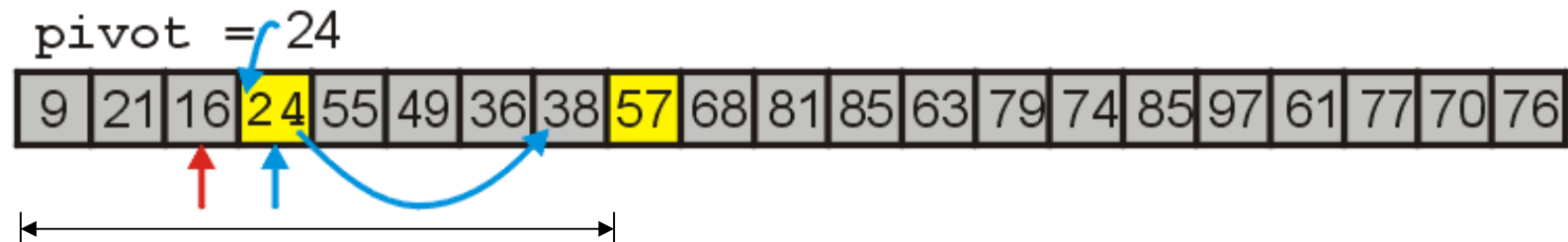
The indices are reversed, so we stop



Quick Sort Example

We move 38 to the vacant location and move the pivot 24 into the location previously occupied by 38

- 24 is now in the correct location



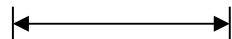
Quick Sort Example

We will now recursively call quick sort on the left and right halves of those entries which are < 57

9	21	16	24	55	49	36	38	57	68	81	85	63	79	74	85	97	61	77	70	76
---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

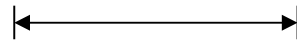
Quick Sort Example

The first partition has three entries, so we sort it using insertion sort



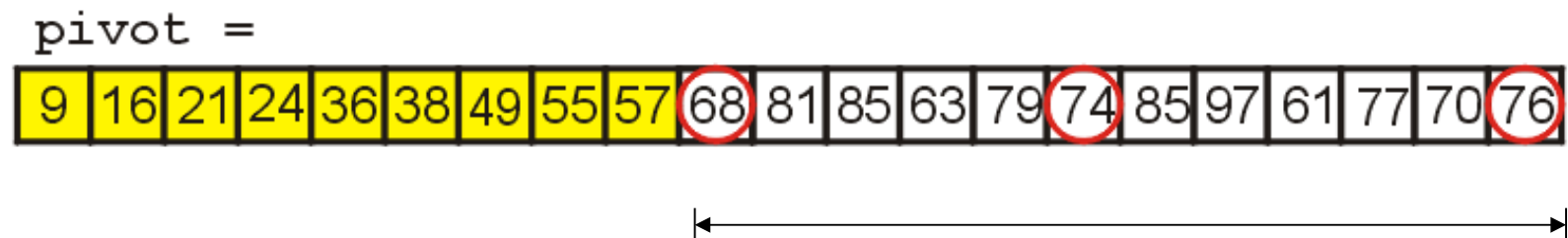
Quick Sort Example

The second partition also has only four entries, so again, we use insertion sort



Quick Sort Example

First we examine the first, middle, and last entries of the sub-list

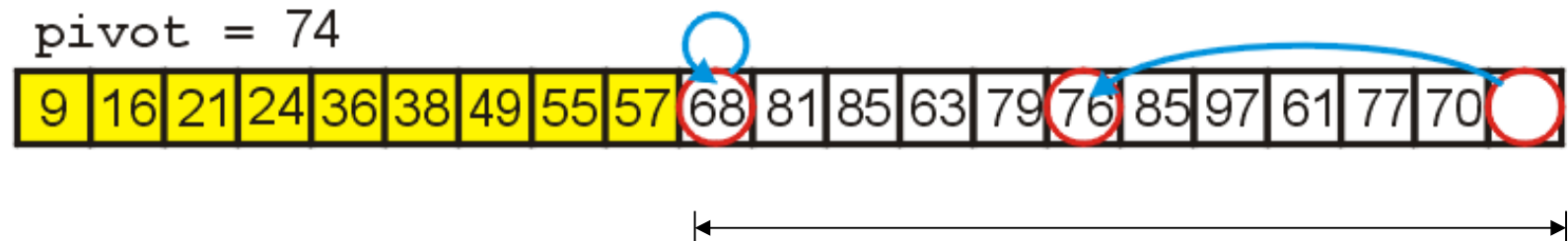


Quick Sort

Quick Sort Example

We choose 74 to be our pivot

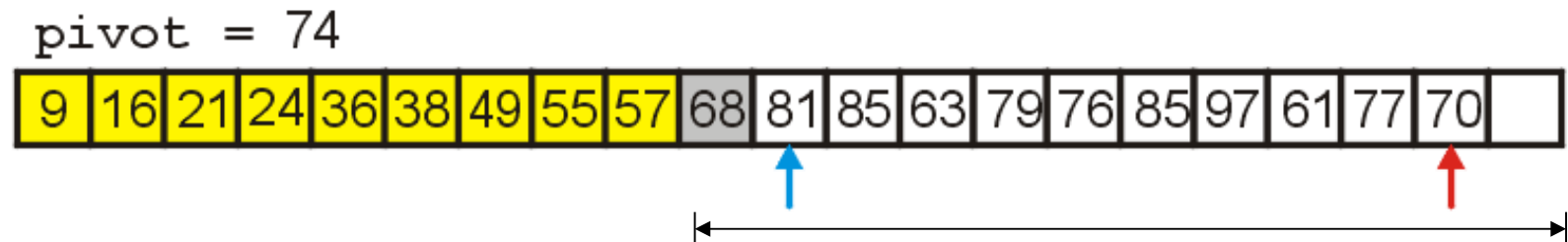
We move 76 to the vacancy left by 74



Quick Sort Example

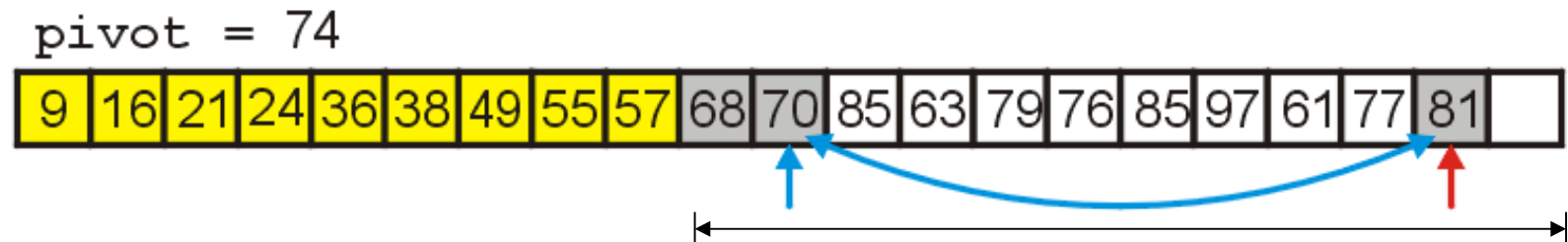
We search forward until we find **81** > 74

We search backward until we find **70** < 74



Quick Sort Example

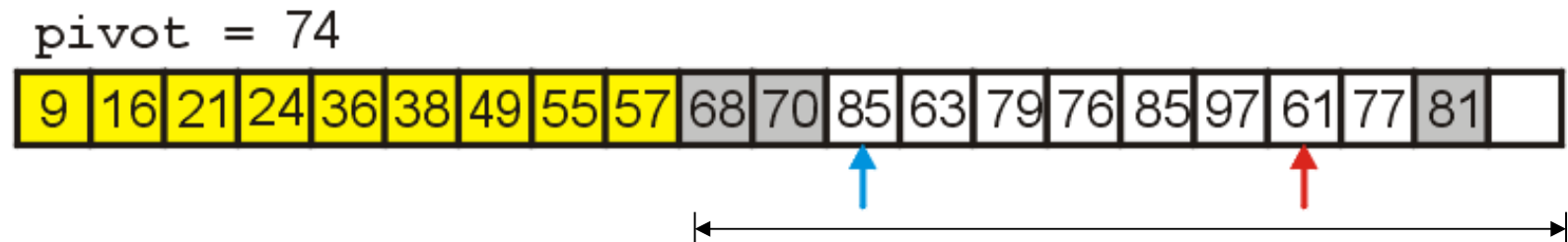
We swap 70 and 84 placing them in order



Quick Sort Example

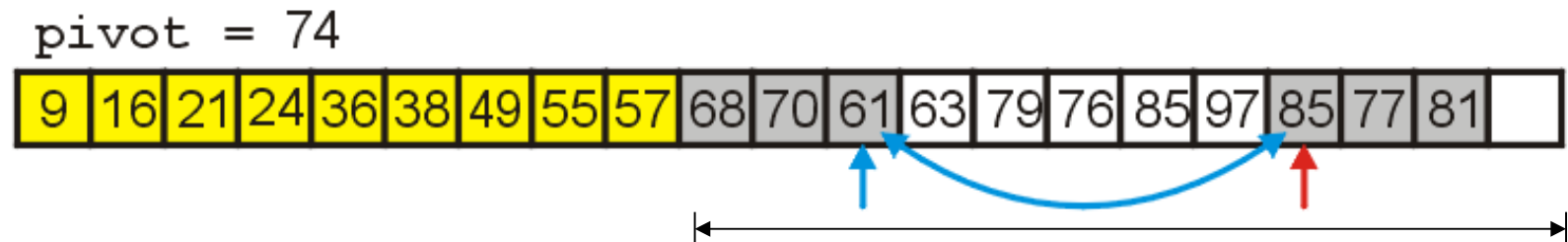
We search forward until we find **85** > 74

We search backward until we find **61** < 74



Quick Sort Example

We swap 85 and 61 placing them in order

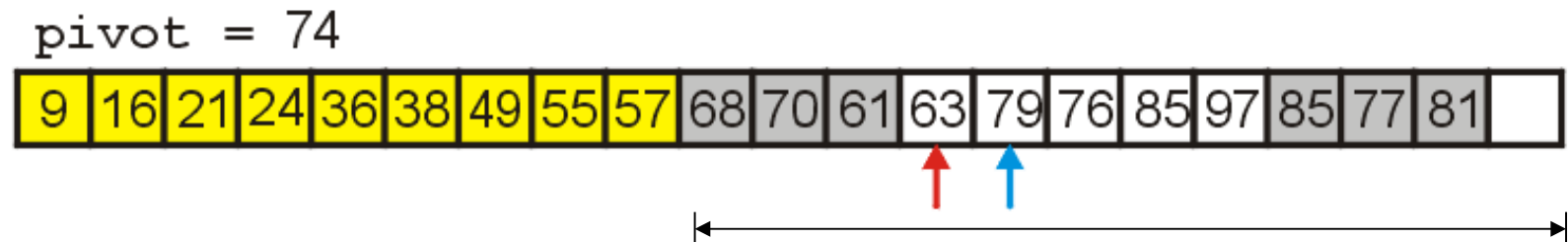


Quick Sort Example

We search forward until we find **79** > 74

We search backward until we find **63** < 74

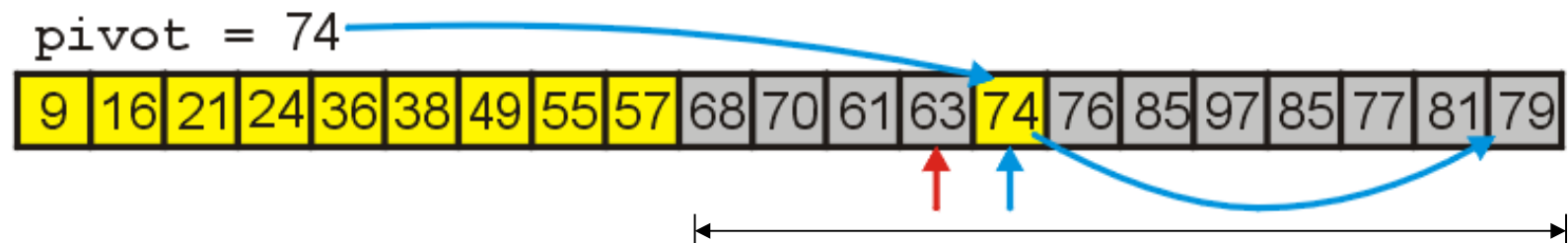
The indices are reversed, so we stop



Quick Sort Example

We move 79 to the vacant location and move the pivot 74 into the location previously occupied by 79

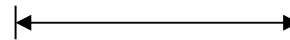
74 is now in the correct location



Quick Sort Example

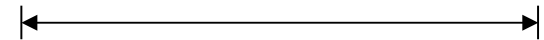
We sort the left sub-list first

It has four elements, so we simply use insertion sort



Quick Sort Example

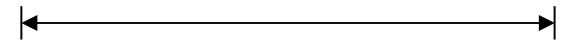
Having sorted the four elements, we focus on the remaining sub-list of seven entries



Quick Sort Example

To sort the next sub-list, we examine the first, middle, and last entries

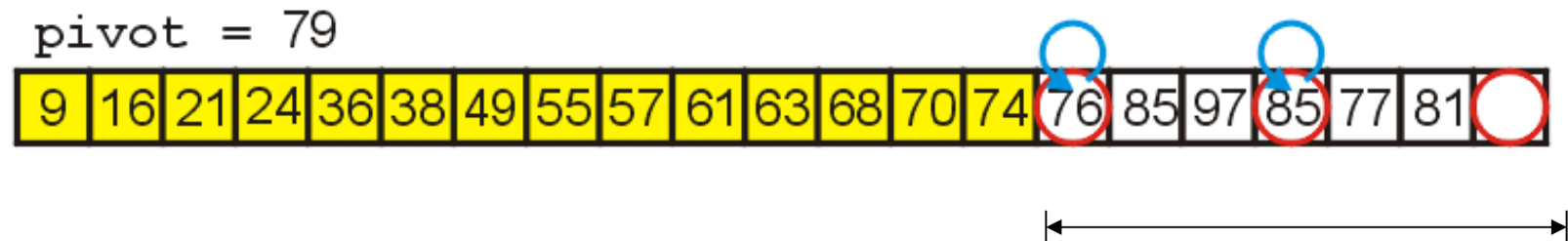
pivot =



Quick Sort Example

We select 79 as our pivot and move:

- 76 into the lowest position
- 85 into the highest position

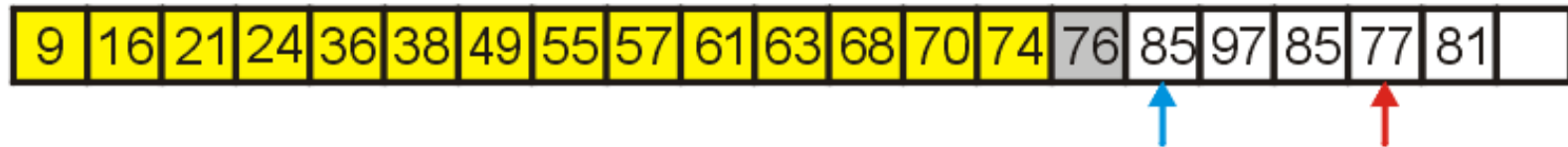


Quick Sort Example

We search forward until we find **85** > 79

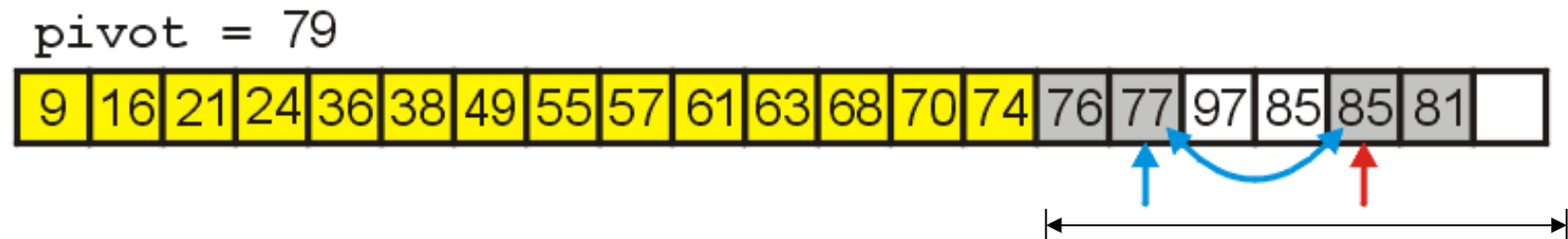
We search backward until we find **77** < 79

pivot = 79



Quick Sort Example

We swap 85 and 77, placing them in order

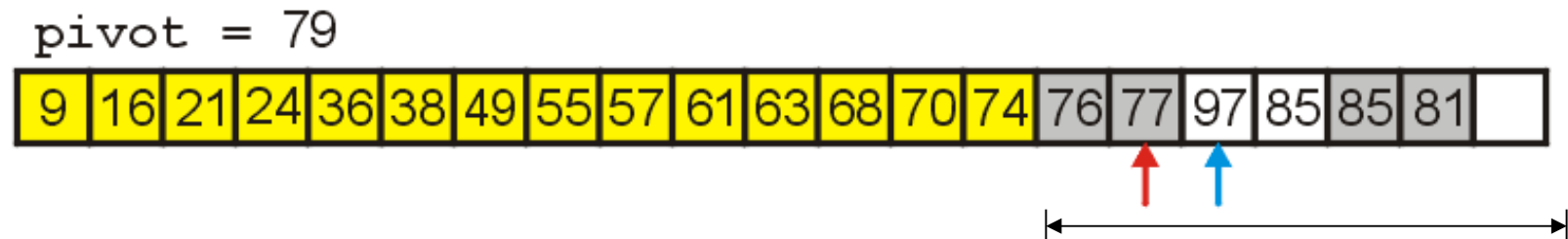


Quick Sort Example

We search forward until we find **97** > 79

We search backward until we find **77** < 79

The indices are reversed, so we stop



Quick Sort Example

Finally, we move 97 to the vacant location and copy 79 into the appropriate location

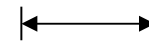
- 79 is now in the correct location



Quick Sort Example

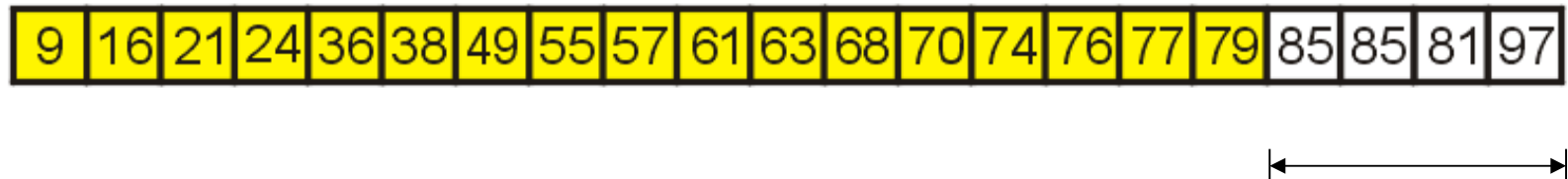
This splits the sub-list into two sub-lists of size 2 and 4

We use insertion sort for the first sub-list



Quick Sort Example

We are left with one sub-list with four entries, so again, we use insertion sort



Quick Sort Example

Sorting the last sub-list, we arrive at an ordered list

9	16	21	24	36	38	49	55	57	61	63	68	70	74	76	77	79	81	85	85	97
---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----