Sorting

Sorting

- A fundamental problem in computing
  - putting a collection of items in order
- Often used as part of another algorithm
  - eg. sort then do many binary searches
  - eg. looking for identical items in array:
    - unsorted: do O(n^2) comparisons
    - sort (O(??)) then scan array and do O(n) comparisons
- There is a `sort()` function in `java.util.Arrays`.

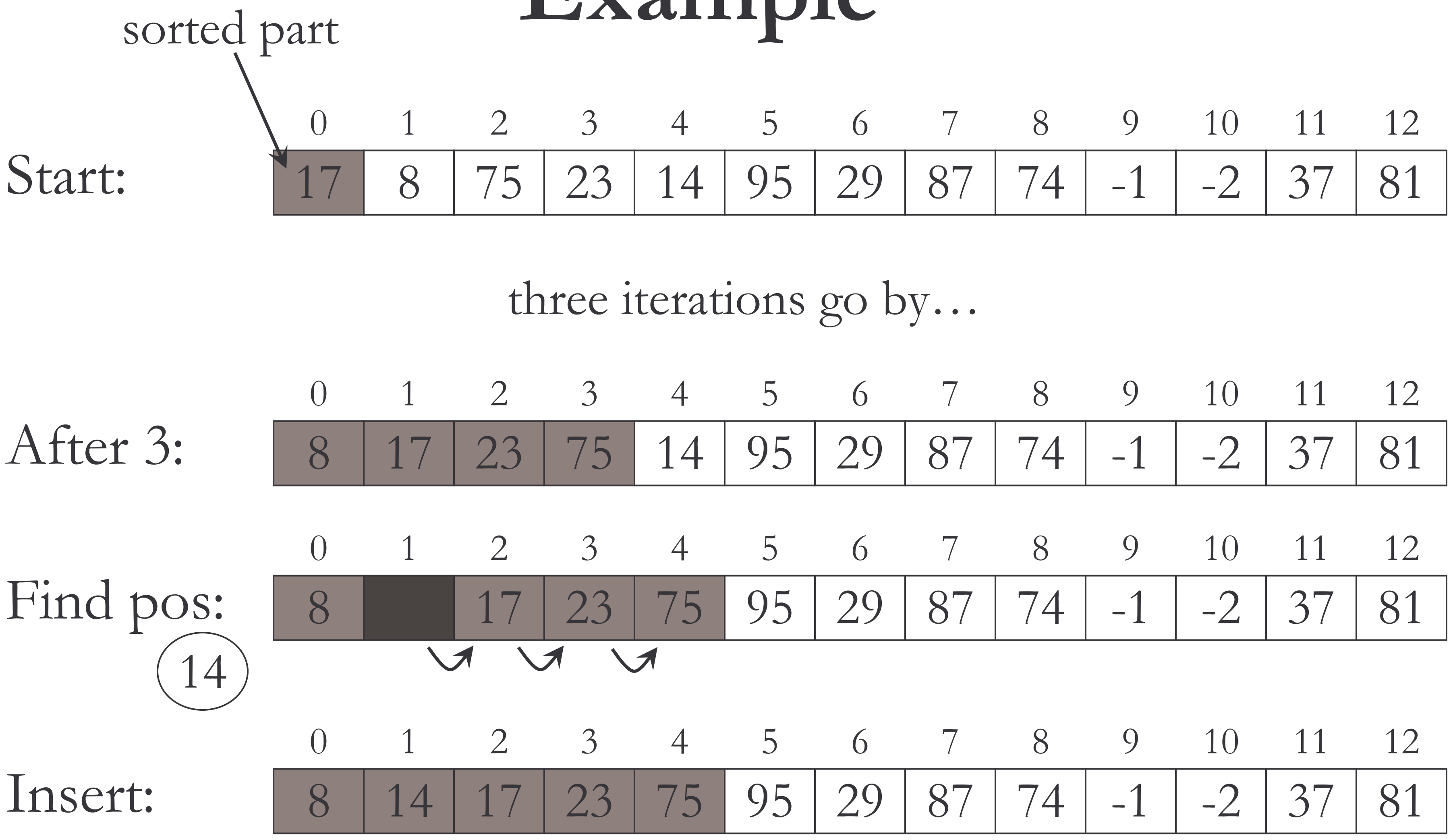
Sorting Algorithms

- There are many algorithms for sorting.
- Each has different properties:
  - easy/hard to understand
  - fast/slow for large lists
  - fast/slow for short lists
  - fast in all cases/on average

Insertion Sort

- The idea: Build a sorted part of the array by moving left to right.
  - take the next item in the list
  - find where it should go (in the sorted part)
  - open up a space for it
  - put it in its place
  - repeat for each item/position

Example



sorted part

	0	1	2	3	4	5	6	7	8	9	10	11	12
Start:	17	8	75	23	14	95	29	87	74	-1	-2	37	81

three iterations go by…

	0	1	2	3	4	5	6	7	8	9	10	11	12
After 3:	8	17	23	75	14	95	29	87	74	-1	-2	37	81

	0	1	2	3	4	5	6	7	8	9	10	11	12
Find pos: 14	8		17	23	75	95	29	87	74	-1	-2	37	81

	0	1	2	3	4	5	6	7	8	9	10	11	12
Insert:	8	14	17	23	75	95	29	87	74	-1	-2	37	81

Pseudocode

```
for pos from 1 to n-1:
    val = array[pos]  // get element pos into place
    i = pos-1
    while $i \geq 0$ and array[i] > val:
        array[i+1] = array[i]
        i--
    array[i+1] = val
```

Example

assume we’ve done pos = 1, 2, 3…

0	1	2	3	4	5	6	7	8	9	10	11	12
8	~~17~~	~~23~~	~~75~~	~~14~~	95	29	87	74	-1	-2	37	81
	14	17	23	75								

pos = 4

val = 14

i = ~~3~~ ~~2~~ ~~1~~ 0

Speed

- requires n-1 passes through the array
  - pass i must compare and move up to i elements
  - Total **maximum** comparisons/moves:

    $1 + 2 + \ldots + (n\text{-}1) = n(n\text{-}1)/2 = n^2/2 - n/2$
- So, insertion sort is order n-squared: $O(n^2)$
  - not bad, but there are much faster algorithms
  - turns out: insertion sort is generally faster than other algorithms for small arrays (maybe n<10?)

“Sorting out Sorting”

- 30 min film on sorting algorithms
- Explanation of various sorting algorithms
- Good overview of the ideas the algorithms are based on.
    - Don’t worry too much about the details.
    - Tree Sort and Heap Sort require knowledge of tree data structures: don’t worry if you don’t understand them.