Insertion Sort

Insertion Sort

- In insertion sort, the idea of the algorithm is to build your sorted array in place, shifting elements out of the way if necessary to make room as you go.

In pseudocode:

- Call the first element of the array “sorted.”
- Repeat until all elements are sorted:
  - Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

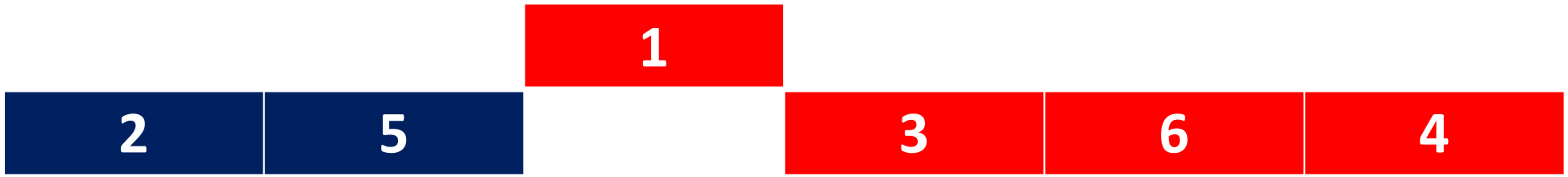
In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort



In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

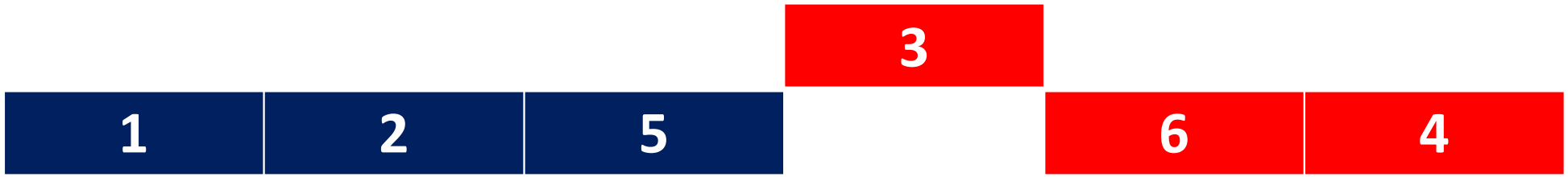
In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort



In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

In pseudocode:

Call the first element of the array “sorted.”

Repeat until all elements are sorted:

Look at the next unsorted element. Insert into the “sorted” portion by shifting the requisite number of elements.

Insertion Sort

- **Worst-case scenario**: The array is in reverse order; we have to shift each of the *n* elements *n* positions each time we make an insertion.

- **Best-case scenario**: The array is already perfectly sorted, and we simply keep moving the line between “unsorted” and “sorted” as we examine each element.

Insertion Sort

$O(n^2)$

$\Omega(n)$