Insertion Sort Algorithm - Step by Step Solution

Given Array:

18, 16, 85, 5, 45, 60, 50, 8, 2, 7

Insertion Sort Process:

Insertion sort builds the final sorted array one item at a time. It works by taking one element from the unsorted part and placing it in its correct position in the sorted part.

Initial State:

- We consider the first element (18) as already sorted
- Sorted portion: [18]
- Unsorted portion: [16, 85, 5, 45, 60, 50, 8, 2, 7]

Iteration 1: Insert 16 into the sorted portion

- Current element to insert: 16
- Compare 16 with 18: 16 < 18, so shift 18 right
- Insert 16 at the correct position
- Array after iteration: [16, 18, 85, 5, 45, 60, 50, 8, 2, 7]
- Sorted portion: [16, 18]
- Unsorted portion: [85, 5, 45, 60, 50, 8, 2, 7]

Iteration 2: Insert 85 into the sorted portion

- Current element to insert: 85
- Compare 85 with 18: 85 > 18, so no need to shift
- Array after iteration: [16, 18, 85, 5, 45, 60, 50, 8, 2, 7]
- Sorted portion: [16, 18, 85]
- Unsorted portion: [5, 45, 60, 50, 8, 2, 7]

Iteration 3: Insert 5 into the sorted portion

- Current element to insert: 5
- Compare 5 with 85: 5 < 85, so shift 85 right
- Compare 5 with 18: 5 < 18, so shift 18 right

- Compare 5 with 16: 5 < 16, so shift 16 right
- Insert 5 at the correct position
- Array after iteration: [5, 16, 18, 85, 45, 60, 50, 8, 2, 7]
- Sorted portion: [5, 16, 18, 85]
- Unsorted portion: [45, 60, 50, 8, 2, 7]

Iteration 4: Insert 45 into the sorted portion

- Current element to insert: 45
- Compare 45 with 85: 45 < 85, so shift 85 right
- Compare 45 with 18: 45 > 18, so no need to shift
- Insert 45 at the correct position
- Array after iteration: [5, 16, 18, 45, 85, 60, 50, 8, 2, 7]
- Sorted portion: [5, 16, 18, 45, 85]
- Unsorted portion: [60, 50, 8, 2, 7]

Iteration 5: Insert 60 into the sorted portion

- Current element to insert: 60
- Compare 60 with 85: 60 < 85, so shift 85 right
- Compare 60 with 45: 60 > 45, so no need to shift
- Insert 60 at the correct position
- Array after iteration: [5, 16, 18, 45, 60, 85, 50, 8, 2, 7]
- Sorted portion: [5, 16, 18, 45, 60, 85]
- Unsorted portion: [50, 8, 2, 7]

Iteration 6: Insert 50 into the sorted portion

- Current element to insert: 50
- Compare 50 with 85: 50 < 85, so shift 85 right
- Compare 50 with 60: 50 < 60, so shift 60 right
- Compare 50 with 45: 50 > 45, so no need to shift
- Insert 50 at the correct position
- Array after iteration: [5, 16, 18, 45, 50, 60, 85, 8, 2, 7]
- Sorted portion: [5, 16, 18, 45, 50, 60, 85]

- Unsorted portion: [8, 2, 7]

Iteration 7: Insert 8 into the sorted portion

- Current element to insert: 8
- Compare 8 with 85: 8 < 85, so shift 85 right
- Compare 8 with 60: 8 < 60, so shift 60 right
- Compare 8 with 50: 8 < 50, so shift 50 right
- Compare 8 with 45: 8 < 45, so shift 45 right
- Compare 8 with 18: 8 < 18, so shift 18 right
- Compare 8 with 16: 8 < 16, so shift 16 right
- Compare 8 with 5: 8 > 5, so no need to shift
- Insert 8 at the correct position
- Array after iteration: [5, 8, 16, 18, 45, 50, 60, 85, 2, 7]
- Sorted portion: [5, 8, 16, 18, 45, 50, 60, 85]
- Unsorted portion: [2, 7]

Iteration 8: Insert 2 into the sorted portion

- Current element to insert: 2
- Compare 2 with 85: 2 < 85, so shift 85 right
- Compare 2 with 60: 2 < 60, so shift 60 right
- Compare 2 with 50: 2 < 50, so shift 50 right
- Compare 2 with 45: 2 < 45, so shift 45 right
- Compare 2 with 18: 2 < 18, so shift 18 right
- Compare 2 with 16: 2 < 16, so shift 16 right
- Compare 2 with 8: 2 < 8, so shift 8 right
- Compare 2 with 5: 2 < 5, so shift 5 right
- Insert 2 at the correct position
- Array after iteration: [2, 5, 8, 16, 18, 45, 50, 60, 85, 7]
- Sorted portion: [2, 5, 8, 16, 18, 45, 50, 60, 85]
- Unsorted portion: [7]

Iteration 9: Insert 7 into the sorted portion

- Current element to insert: 7
- Compare 7 with 85: 7 < 85, so shift 85 right
- Compare 7 with 60: 7 < 60, so shift 60 right
- Compare 7 with 50: 7 < 50, so shift 50 right
- Compare 7 with 45: 7 < 45, so shift 45 right
- Compare 7 with 18: 7 < 18, so shift 18 right
- Compare 7 with 16: 7 < 16, so shift 16 right
- Compare 7 with 8: 7 < 8, so shift 8 right
- Compare 7 with 5: 7 > 5, so no need to shift
- Insert 7 at the correct position
- Array after iteration: [2, 5, 7, 8, 16, 18, 45, 50, 60, 85]

Final Sorted Array:

[2, 5, 7, 8, 16, 18, 45, 50, 60, 85]