Analysis and Design of Algorithms

Sorting Algorithms

Table of Contents

Sorting Algorithms

- **Sorting Algorithm** is an algorithm made up of a series of instructions that takes an array as input, and outputs a sorted array.

- There are many sorting algorithms, such as:
  - Selection Sort, Bubble Sort, Insertion Sort, Merge Sort, Heap Sort, QuickSort, Radix Sort, Counting Sort, Bucket Sort, ShellSort, Comb Sort, Pigeonhole Sort, Cycle Sort

Bubble Sort

Bubble Sort

- Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.

Bubble Sort

Algorithm:

- **Step1**: Compare each pair of adjacent elements in the list
- **Step2**: Swap two element if necessary
- **Step3**: Repeat this process for all the elements until the entire array is sorted

Bubble Sort

- Example 1 Assume the following Array:

5	1	4	2

Bubble Sort

- First Iteration:
- Compare

Bubble Sort

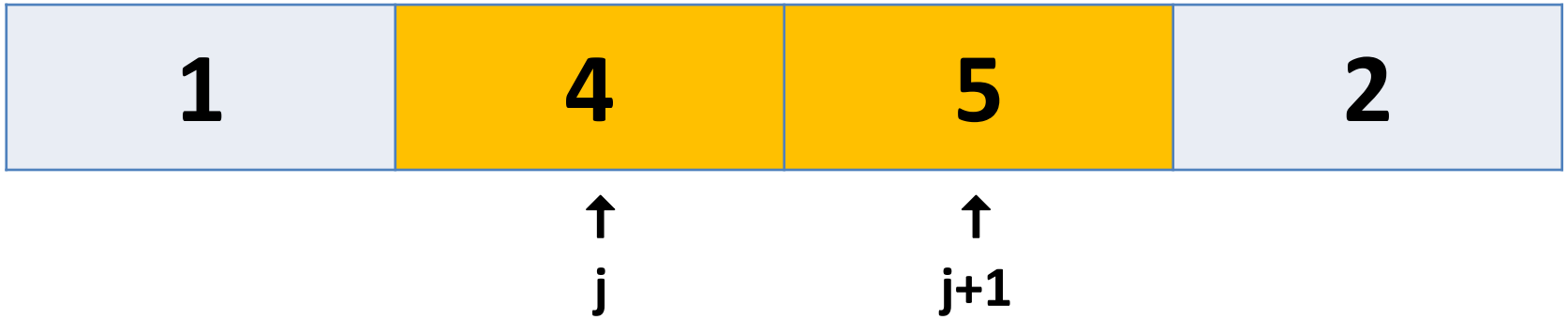
- First Iteration:
- Swap

Bubble Sort

- First Iteration:
- Compare

Bubble Sort

- First Iteration:
- Swap



Bubble Sort

- First Iteration:
- Compare

Bubble Sort

- First Iteration:
- Swap

Bubble Sort

Bubble Sort

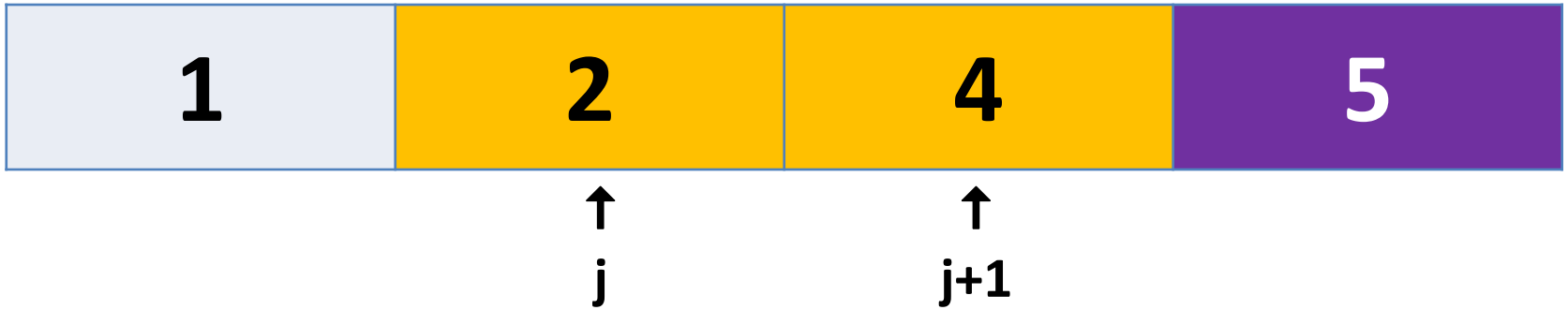
- Second Iteration:
- Compare

Bubble Sort

- Second Iteration:
- Compare

Bubble Sort

- Second Iteration:
- Swap



Bubble Sort

Bubble Sort

- Third Iteration:
- Compare

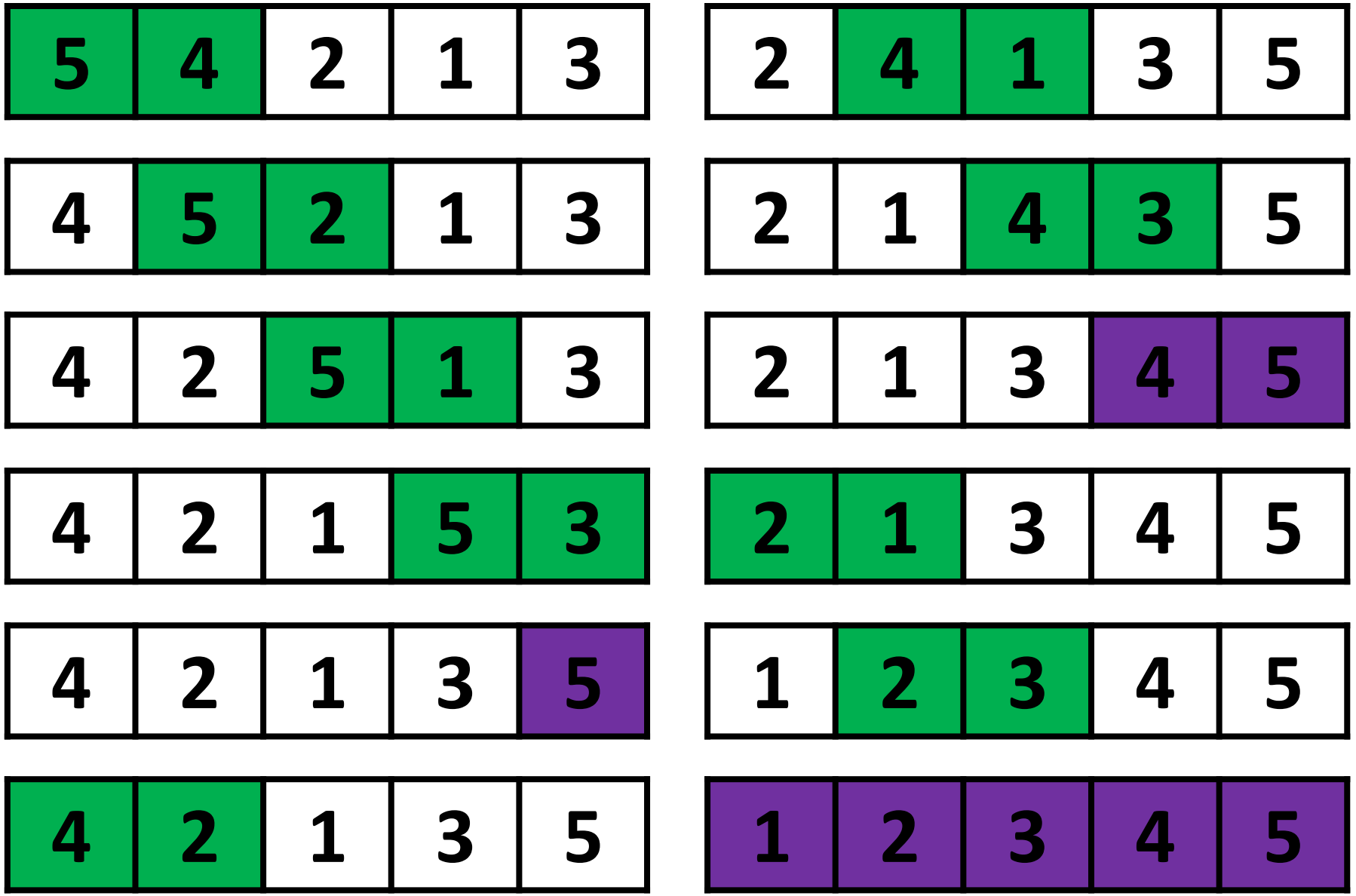
Bubble Sort

Bubble Sort

- Array is now sorted

Bubble Sort

- Example 2:



Bubble Sort

- What is the output of bubble sort after the 1st iteration given the following sequence of numbers: **13 2 9 4 18 45 37 63**

  a) 2 4 9 13 18 37 45 63

  b) 2 9 4 13 18 37 45 63

  c) 13 2 4 9 18 45 37 63

  d) 2 4 9 13 18 45 37 63

Bubble Sort

- What is the output of bubble sort after the 1st iteration given the following sequence of numbers: **13 2 9 4 18 45 37 63**

a) 2 4 9 13 18 37 45 63

b) 2 9 4 13 18 37 45 63

c) 13 2 4 9 18 45 37 63

d) 2 4 9 13 18 45 37 63

Bubble Sort

- **Time Complexity:** $O(n^2)$ as there are two nested loops

- Example of worst case

5	4	3	2	1

Selection Sort

Selection Sort

- The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning.

Selection Sort

Algorithm:

- **Step1**: Find the minimum value in the list
- **Step2**: Swap it with the value in the current position
- **Step3**: Repeat this process for all the elements until the entire array is sorted

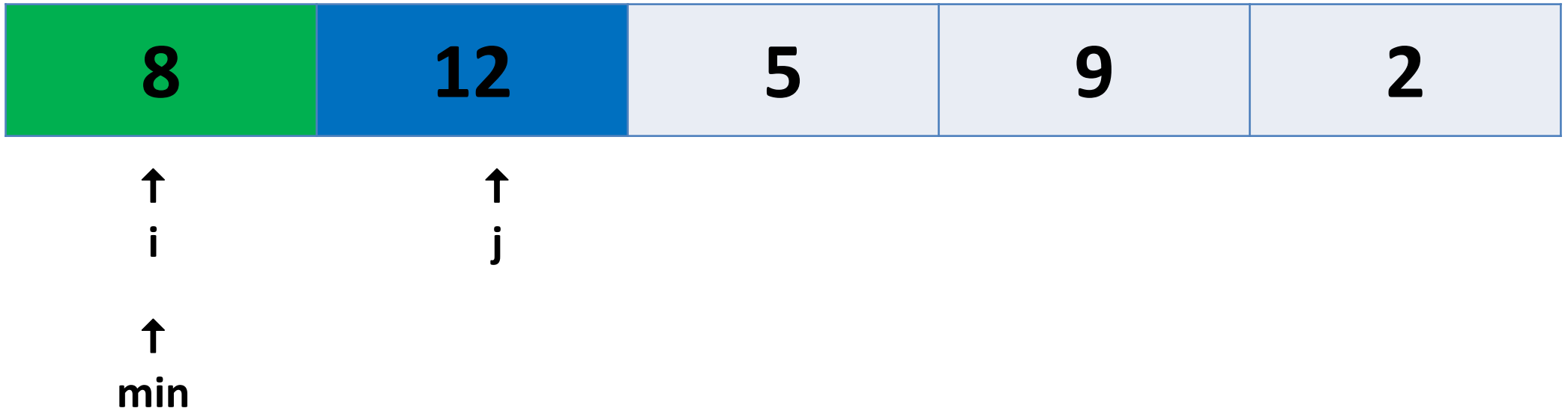
Selection Sort

- Example 1 Assume the following Array:

8	12	5	9	2

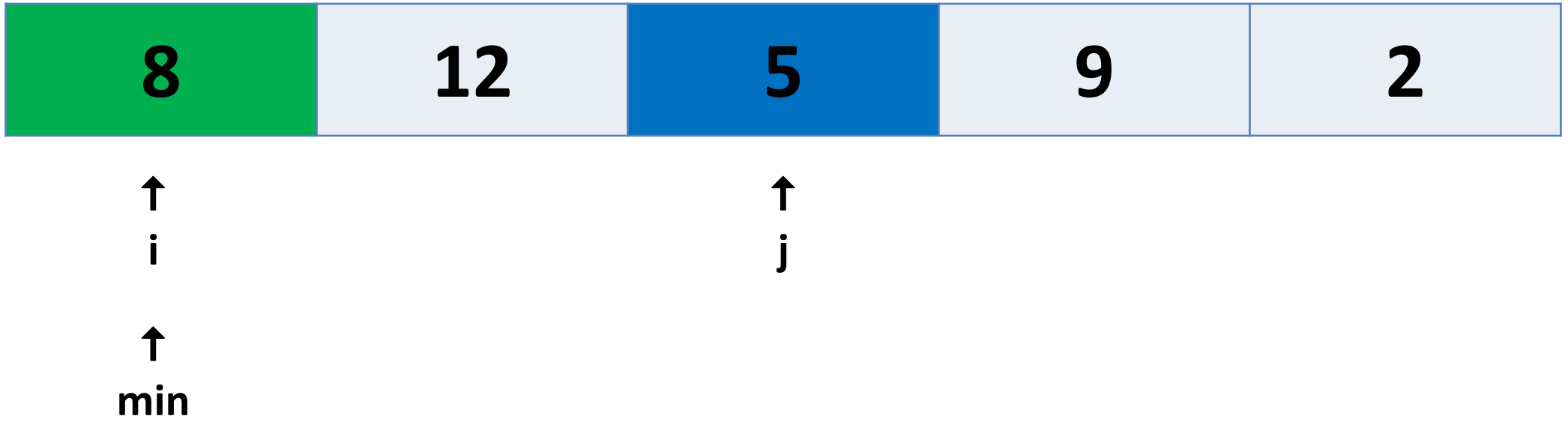
Selection Sort

- Compare



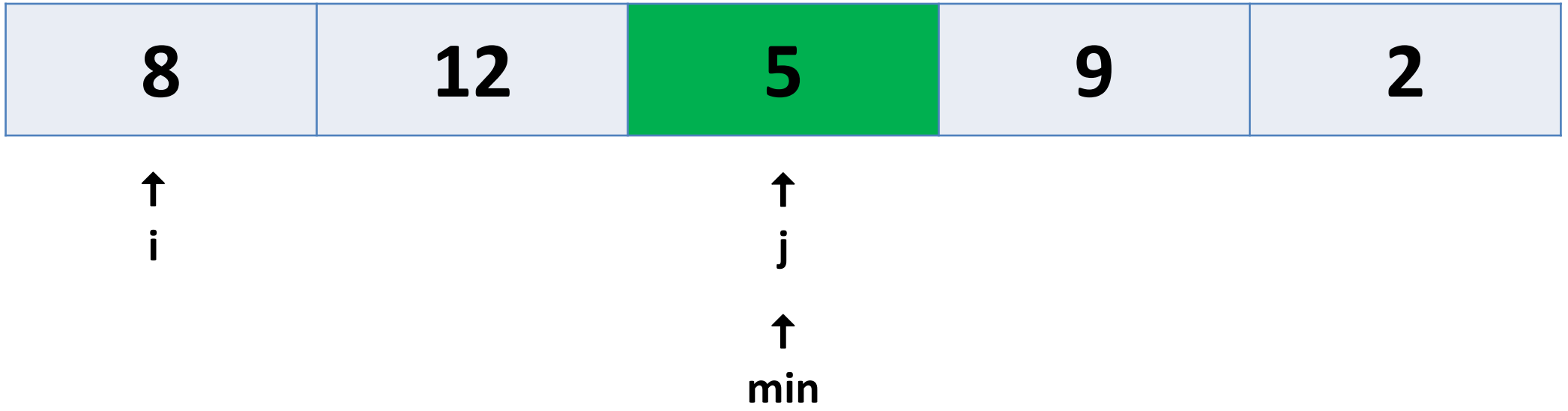
Selection Sort

- Compare



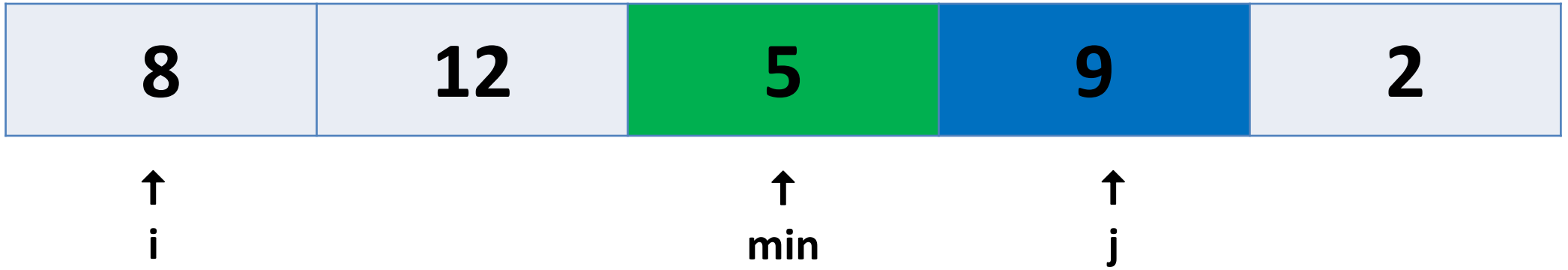
Selection Sort

- Move



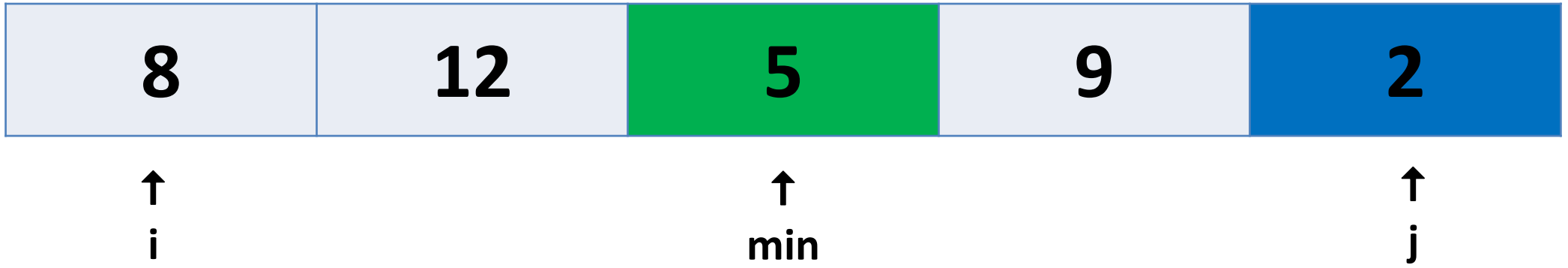
Selection Sort

- Compare



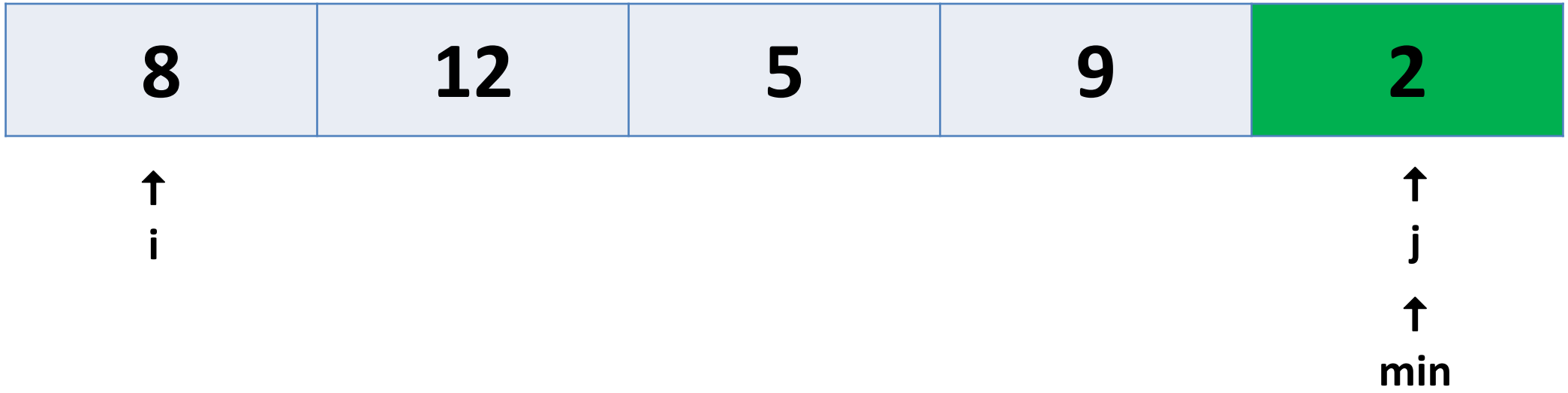
Selection Sort

- Compare



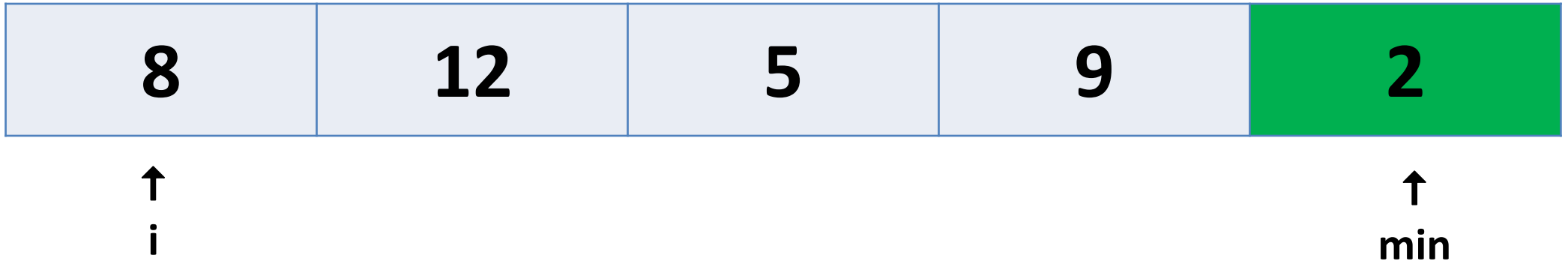
Selection Sort

- Move



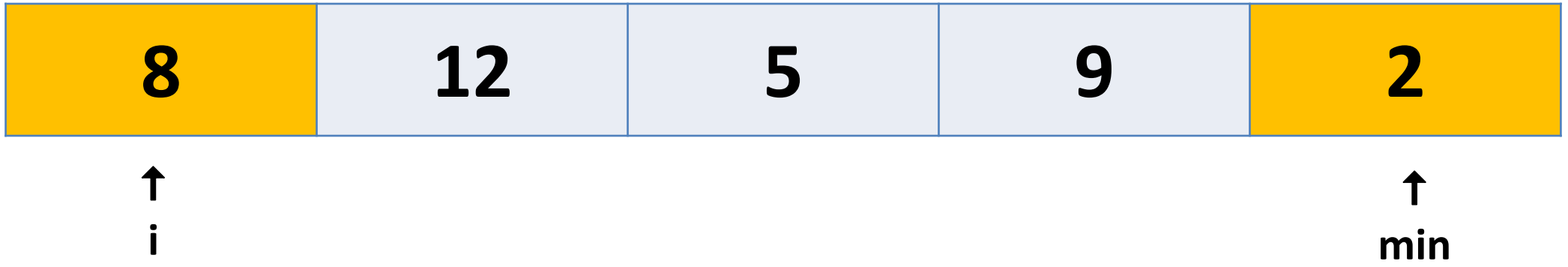
Selection Sort

- Smallest



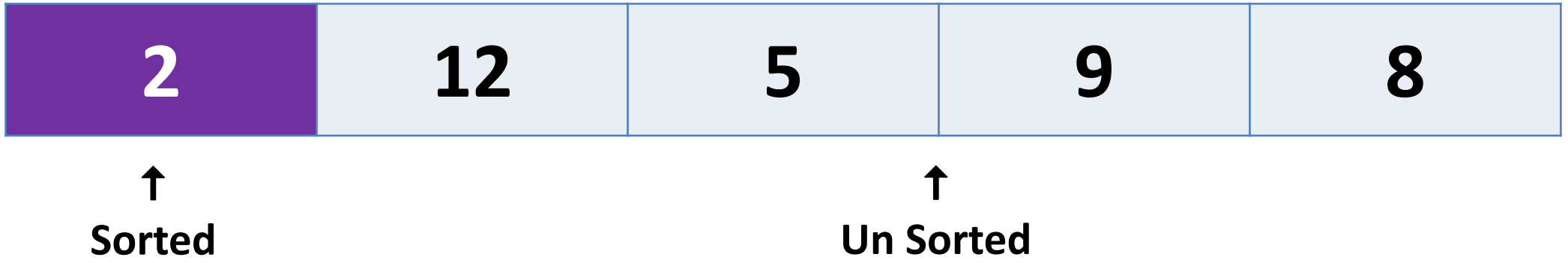
Selection Sort

- Swap



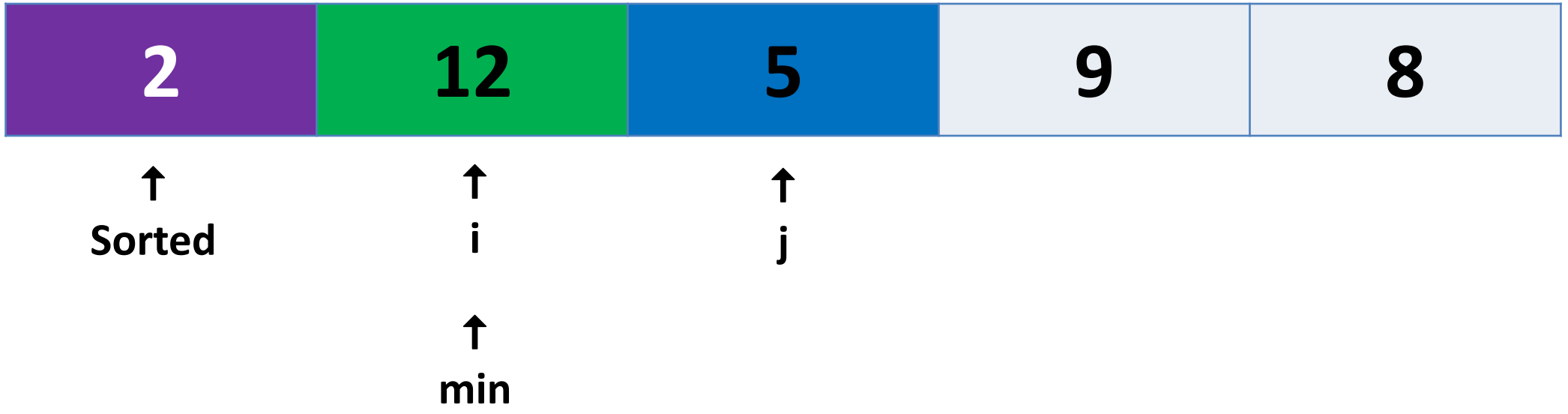
Selection Sort

- Sorted
- Un Sorted



Selection Sort

- Compare

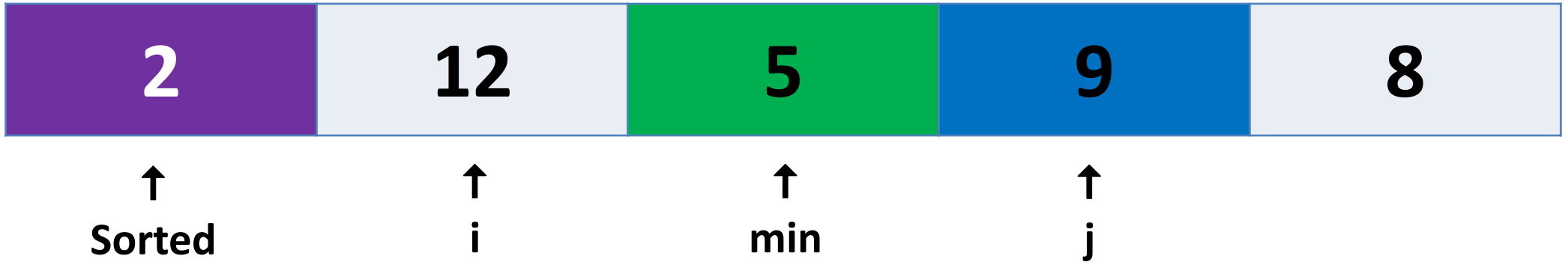


Selection Sort

- Move

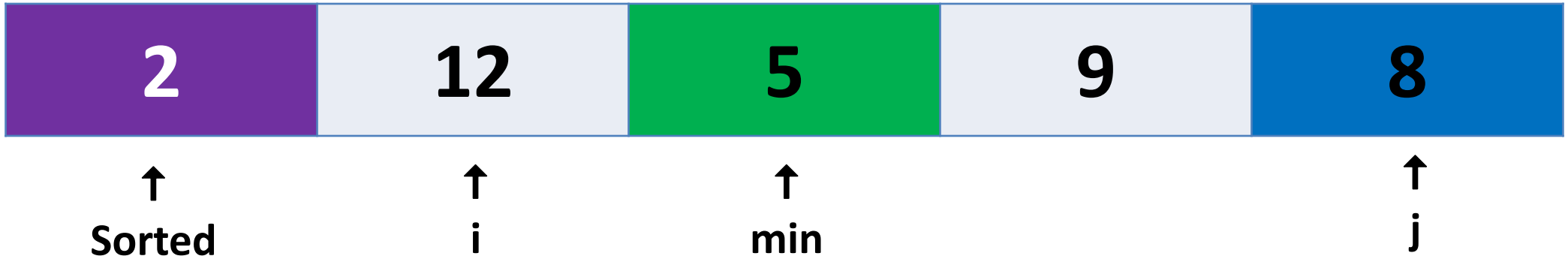
Selection Sort

- Compare



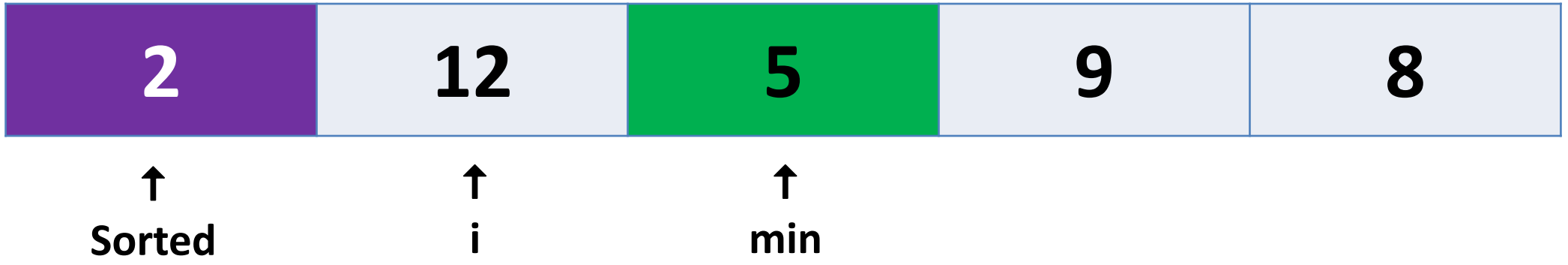
Selection Sort

- Compare



Selection Sort

- Smallest



Selection Sort

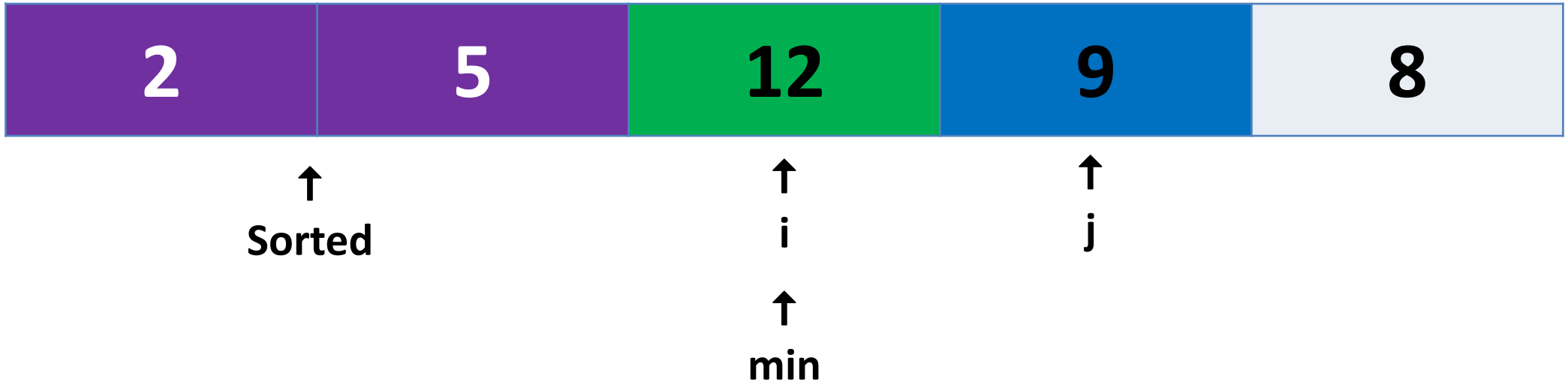
- Swap

Selection Sort

- Sorted
- Un Sorted

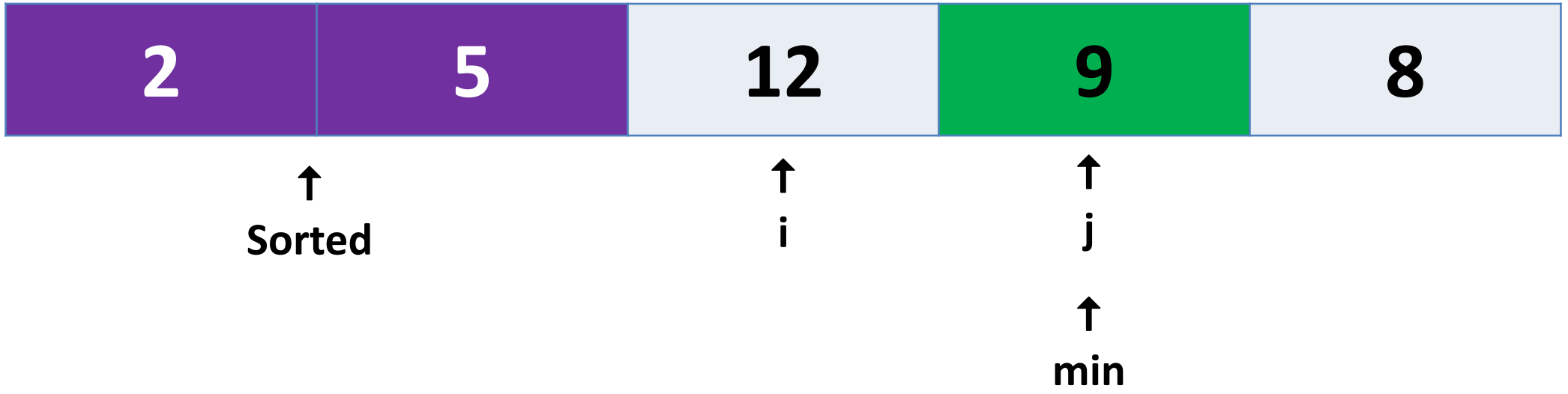
Selection Sort

- Compare



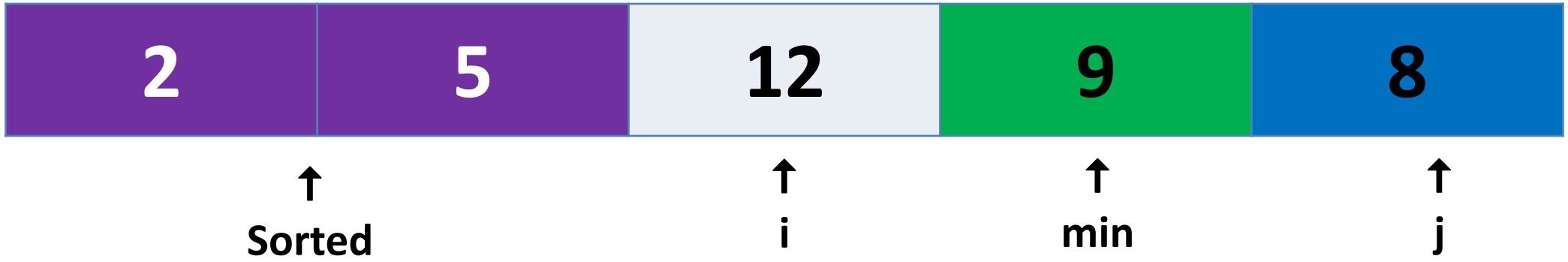
Selection Sort

- Move



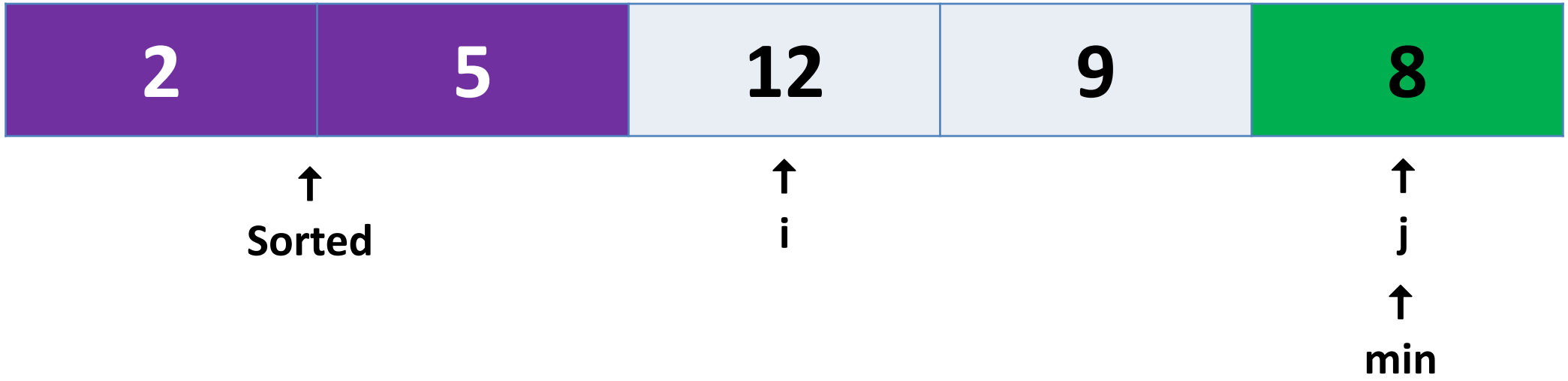
Selection Sort

- Compare



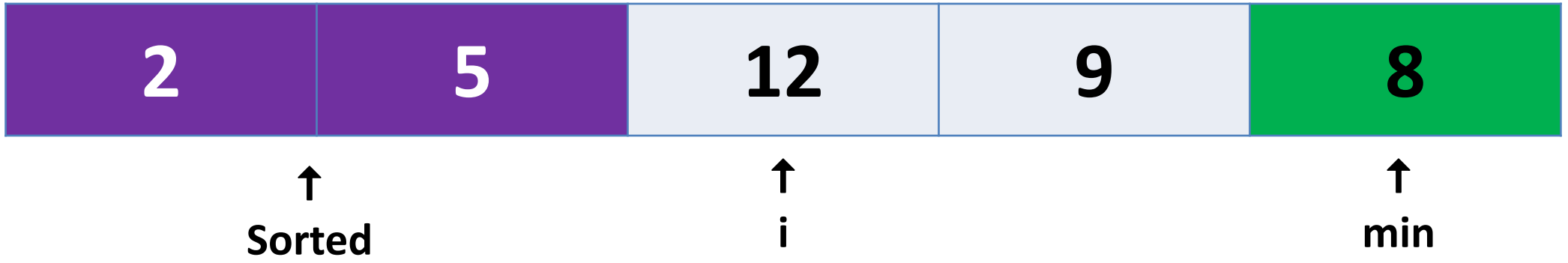
Selection Sort

- Move



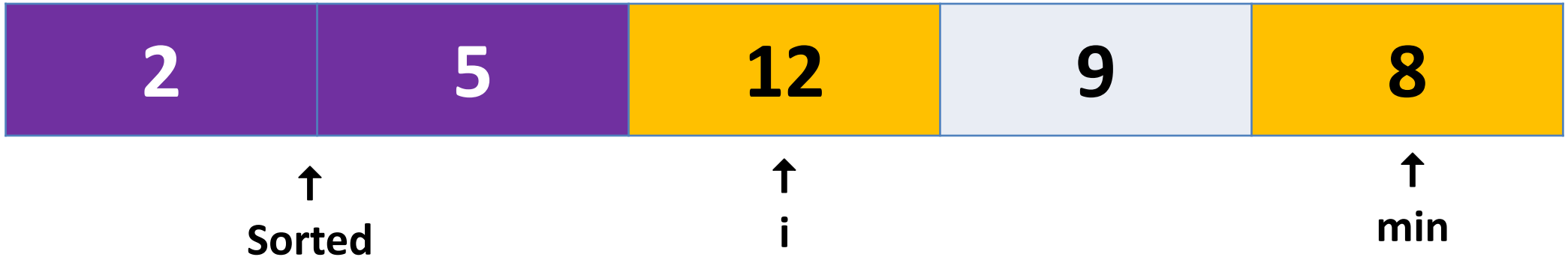
Selection Sort

- Smallest



Selection Sort

- Swap

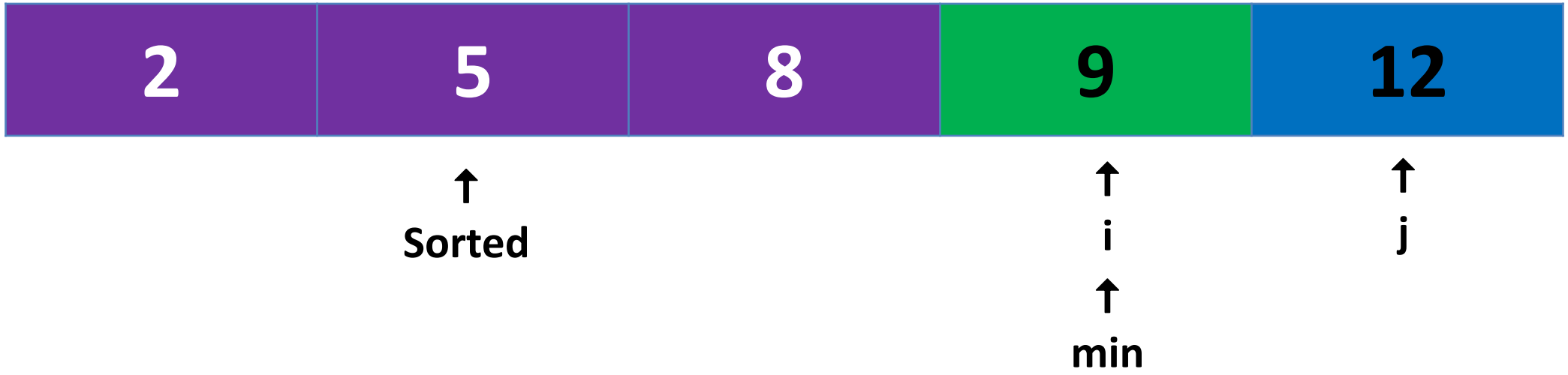


Selection Sort

- Sorted
- Un Sorted

Selection Sort

- Compare

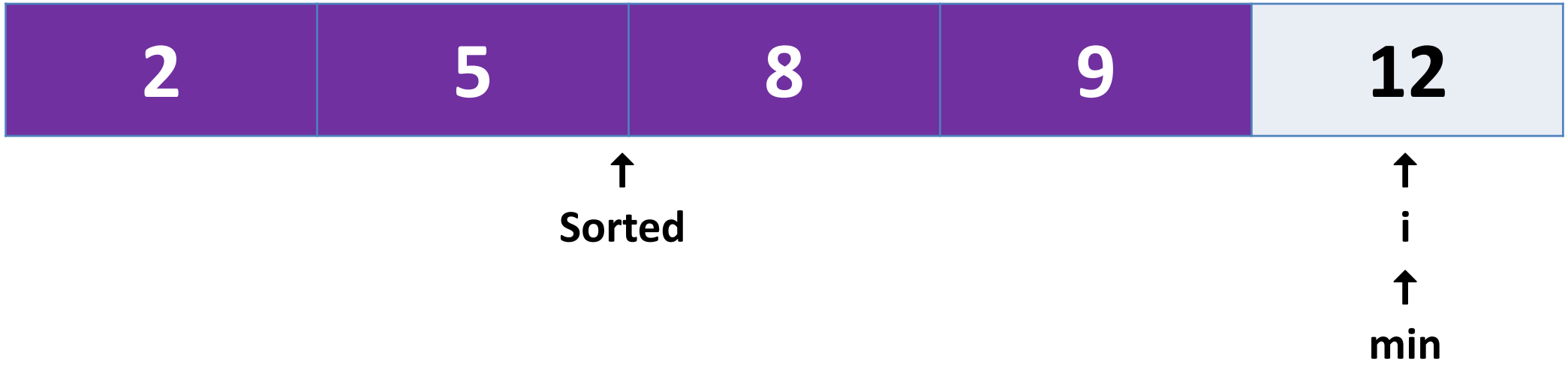


Selection Sort

- Sorted
- Un Sorted

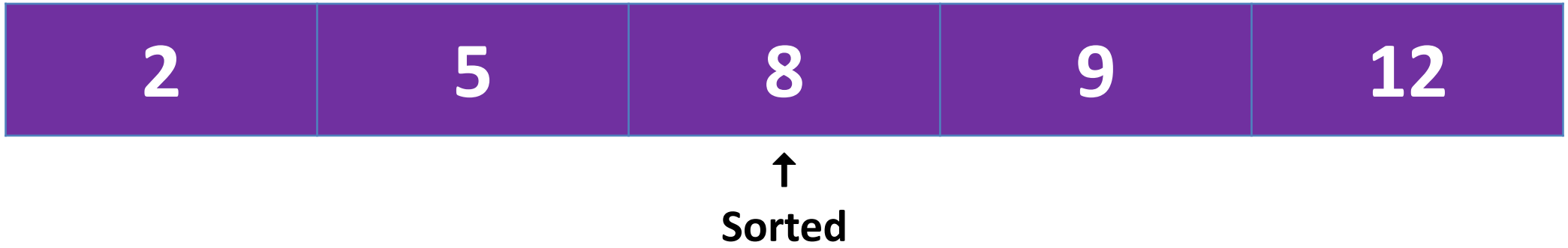
Selection Sort

- Sorted
- Un Sorted



Selection Sort

- Array is now sorted



Selection Sort

- Example 2:

12	10	16	11	9	7

12	10	16	11	9	7
7	10	16	11	9	12
7	9	16	11	10	12
7	9	10	11	16	12
7	9	10	11	16	12
7	9	10	11	12	16

Selection Sort

- What is the output of selection sort after the 2nd iteration given the following sequence of numbers: **13 2 9 4 18 45 37 63**

  a) 2 4 9 13 18 37 45 63

  b) 2 9 4 13 18 37 45 63

  c) 13 2 4 9 18 45 37 63

  d) 2 4 9 13 18 45 37 63

Selection Sort

- What is the output of selection sort after the 2nd iteration given the following sequence of numbers: **13 2 9 4 18 45 37 63**

  a) 2 4 9 13 18 37 45 63

  b) 2 9 4 13 18 37 45 63

  c) 13 2 4 9 18 45 37 63

  d) 2 4 9 13 18 45 37 63

Selection Sort

- **Time Complexity:** $O(n^2)$ as there are two nested loops

- Example of worst case

2	3	4	5	1

Insertion Sort

Insertion Sort

- **Insertion sort** is a simple sorting algorithm that works the way we sort playing cards in our hands.

Insertion Sort

❑ Algorithm:

- **Step1**: Compare each pair of adjacent elements in the list
- **Step2**: Insert element into the sorted list, until it occupies correct position.
- **Step3**: Swap two element if necessary
- **Step4**: Repeat this process for all the elements until the entire array is sorted

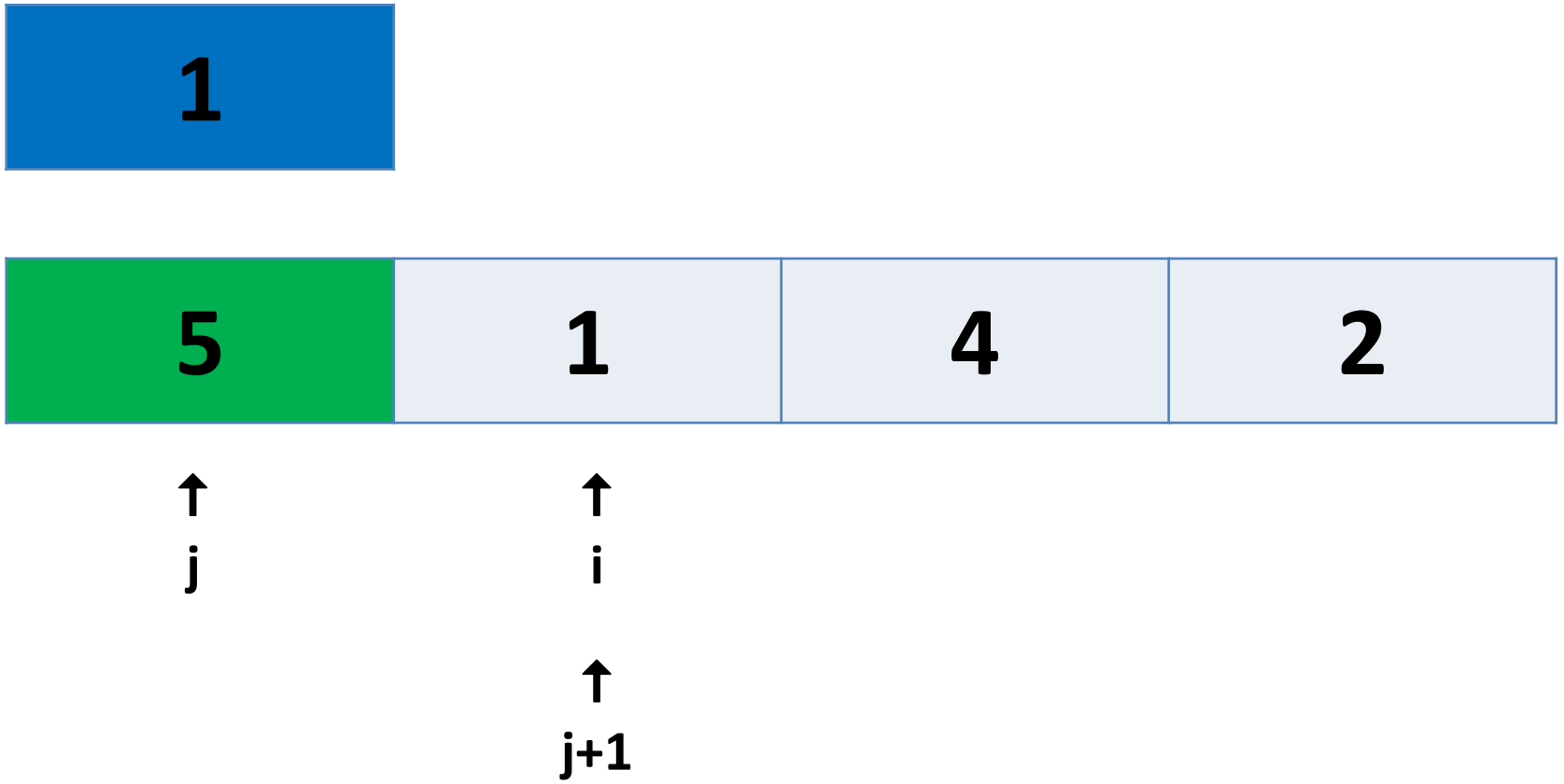
Insertion Sort

- Assume the following Array:

5	1	4	2

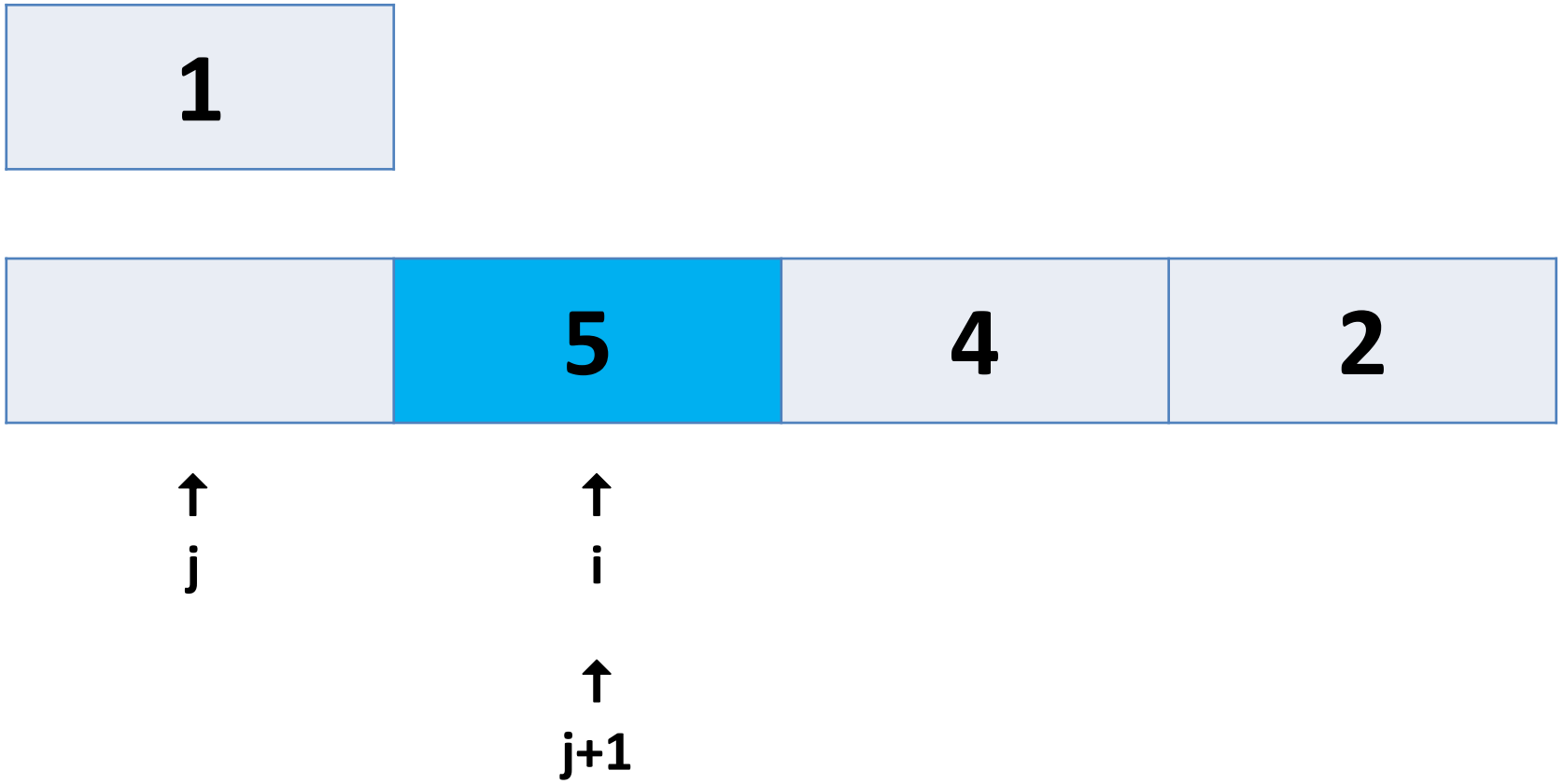
Insertion Sort

- Compare
- Store=



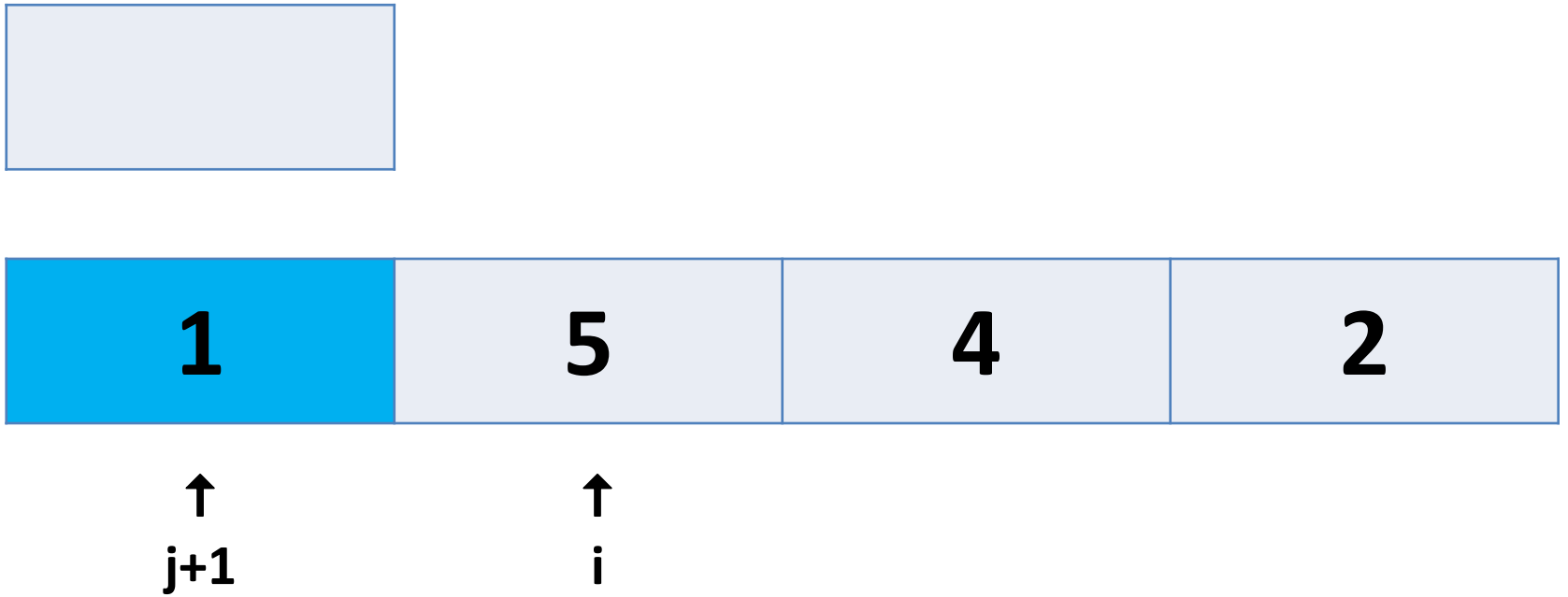
Insertion Sort

- Move
- Store=



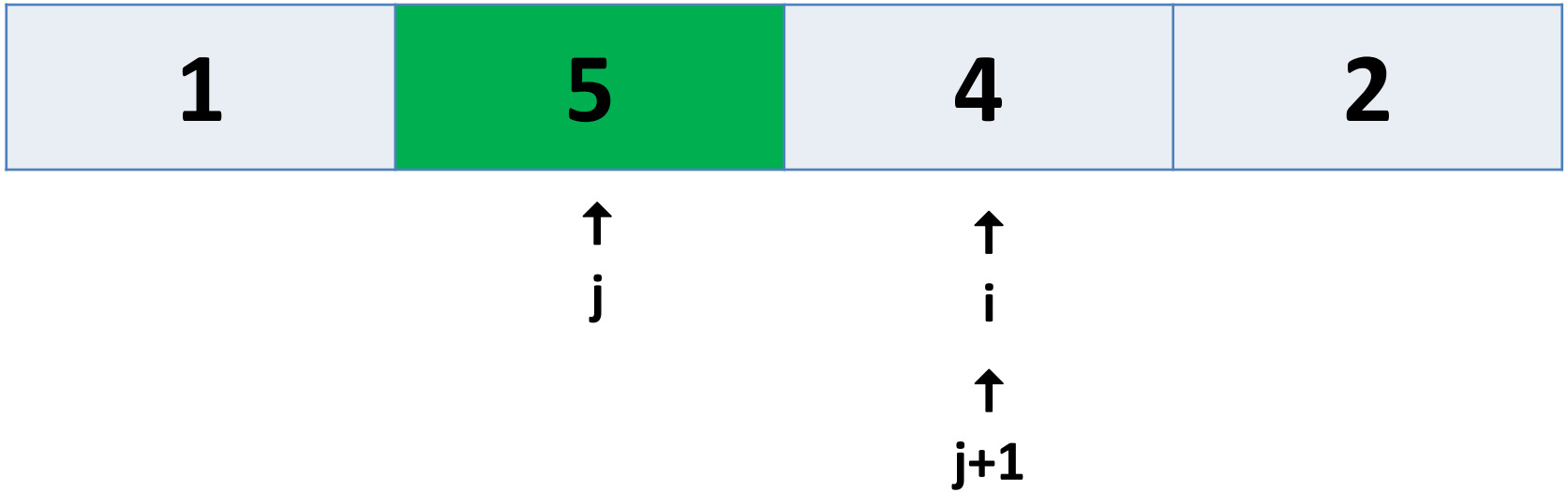
Insertion Sort

- Move
- Store=



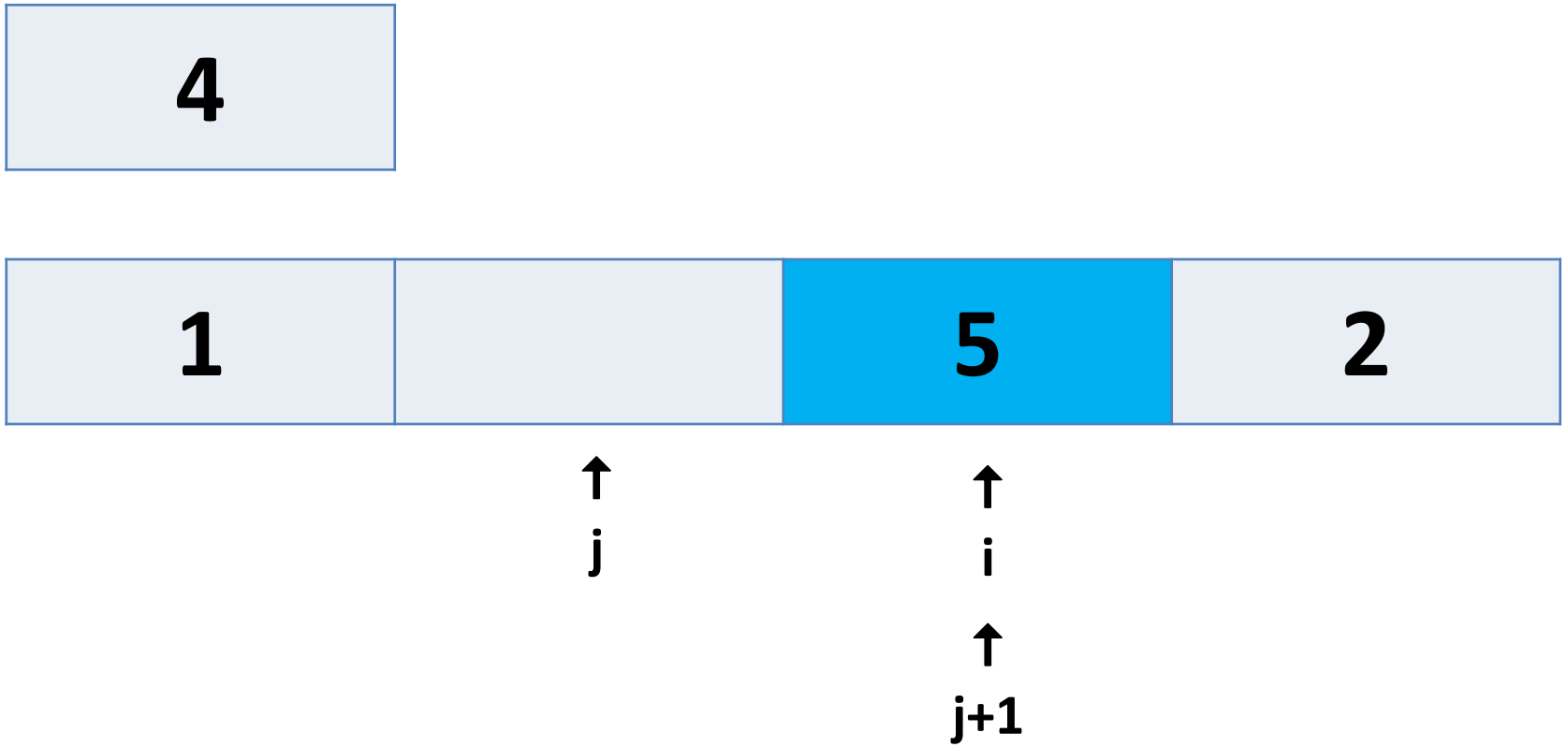
Insertion Sort

- Compare
- Store= 4



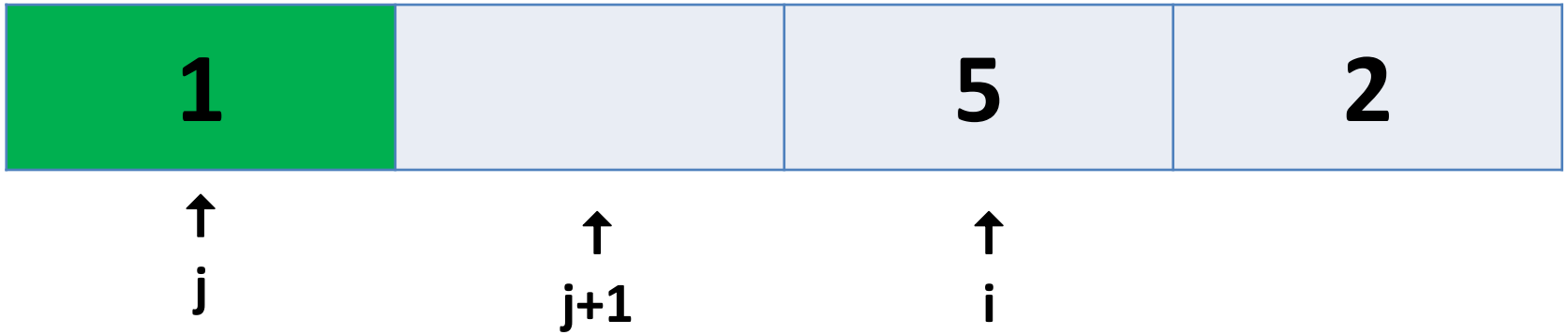
Insertion Sort

- Move
- Store=



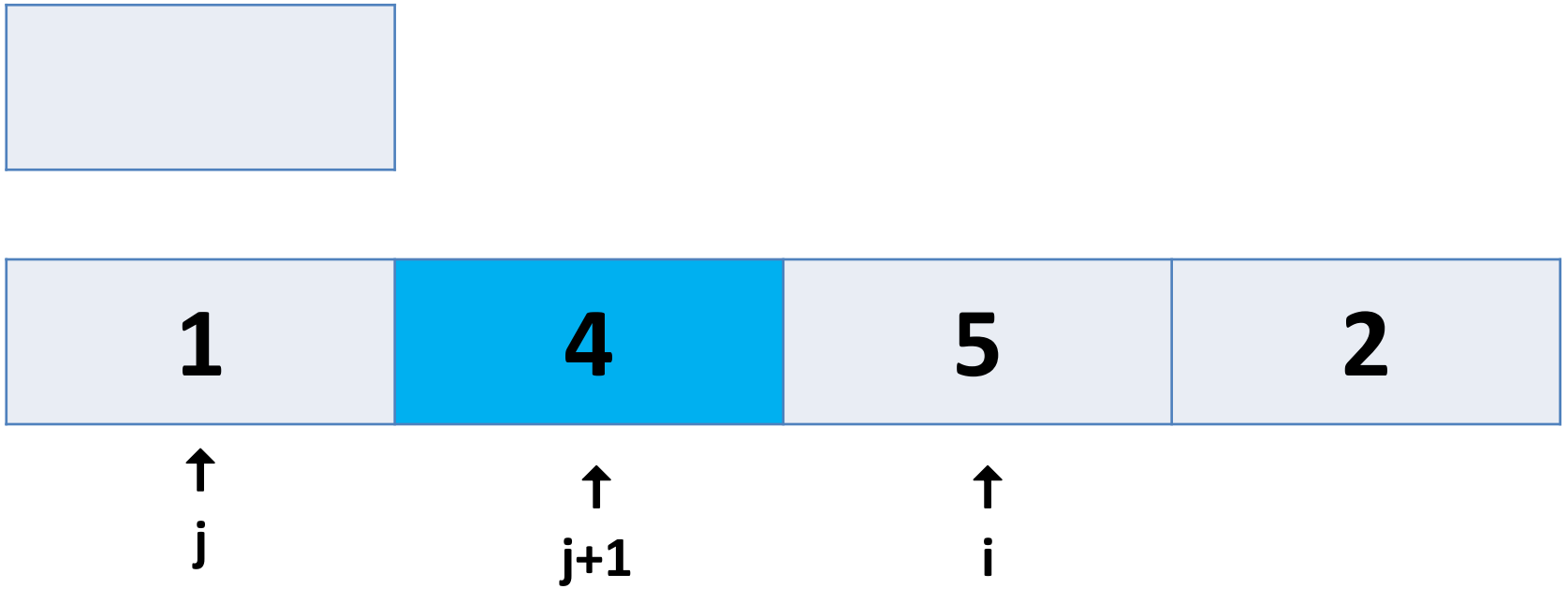
Insertion Sort

- Compare
- Store= 4



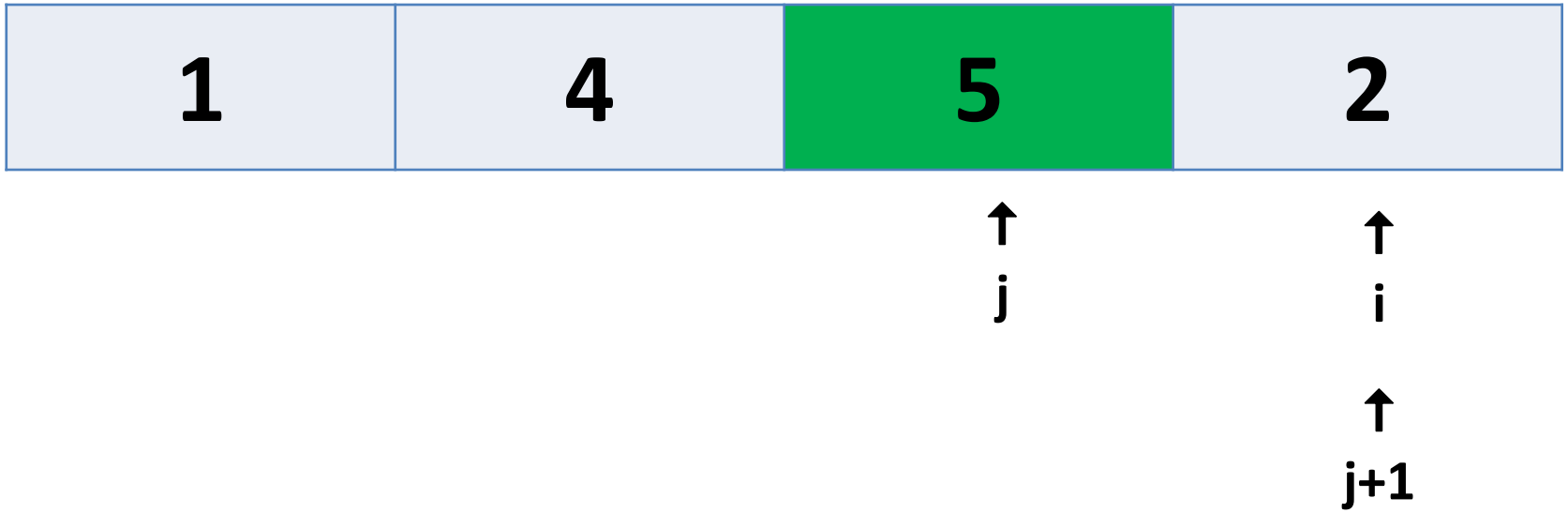
Insertion Sort

- Move
- Store=



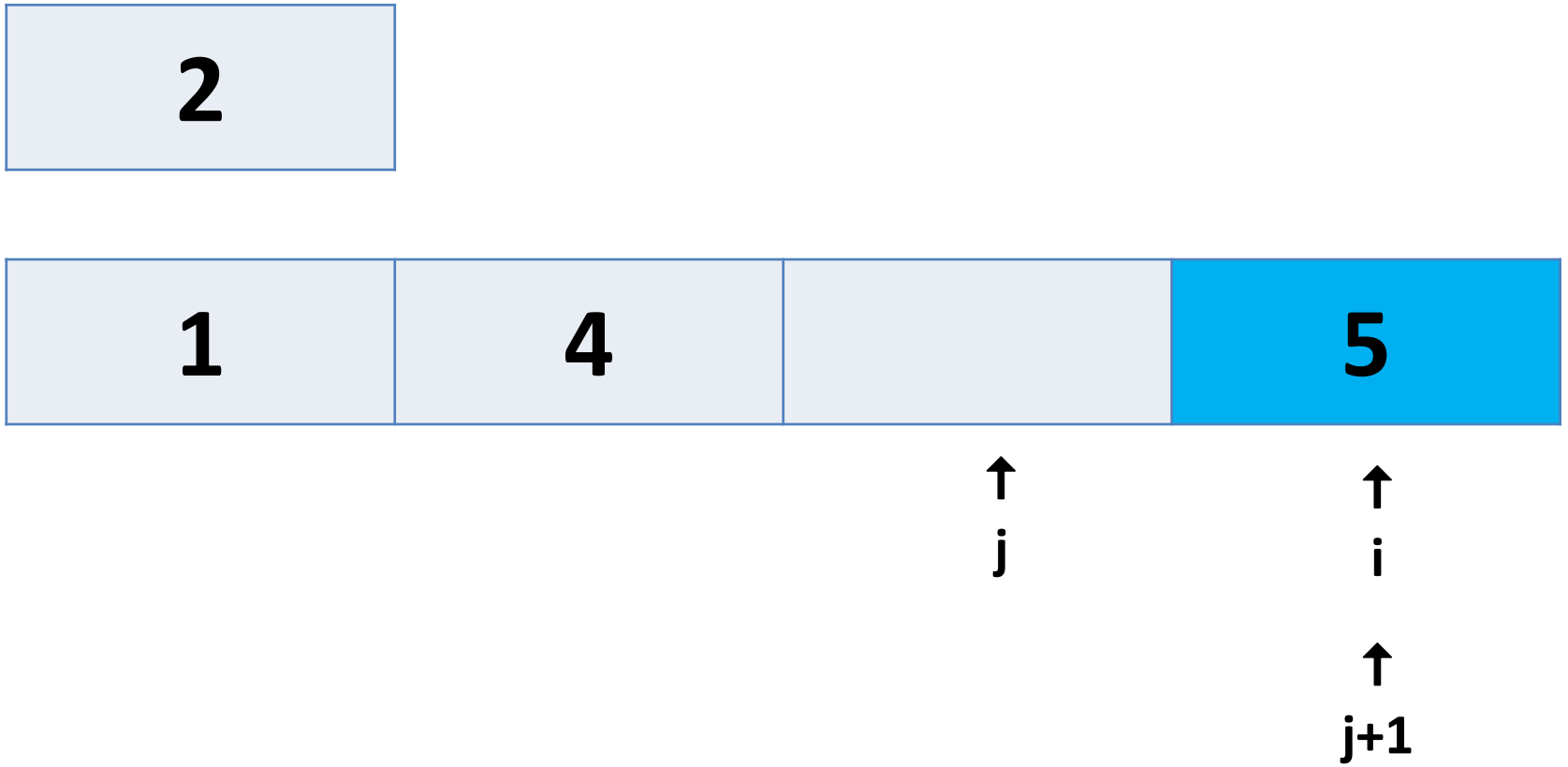
Insertion Sort

- Compare
- Store= **2**



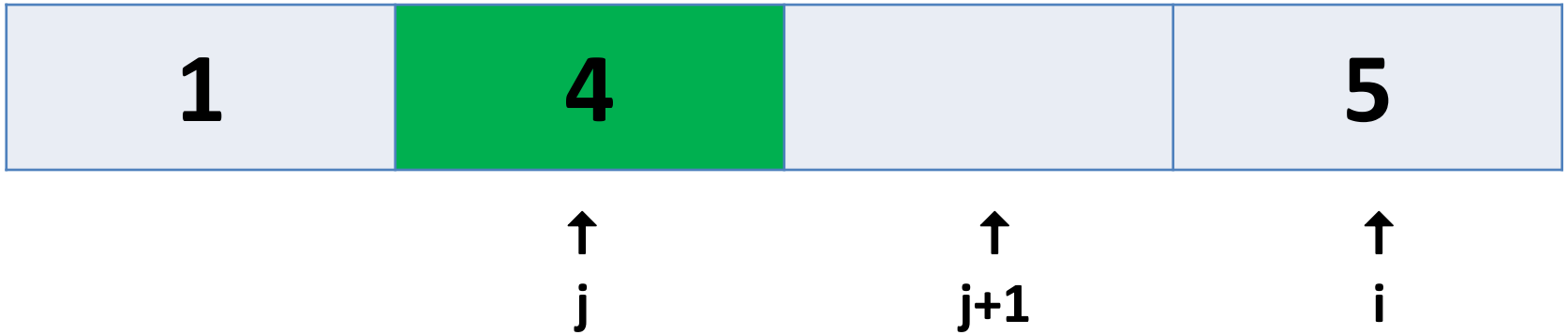
Insertion Sort

- Move
- Store=



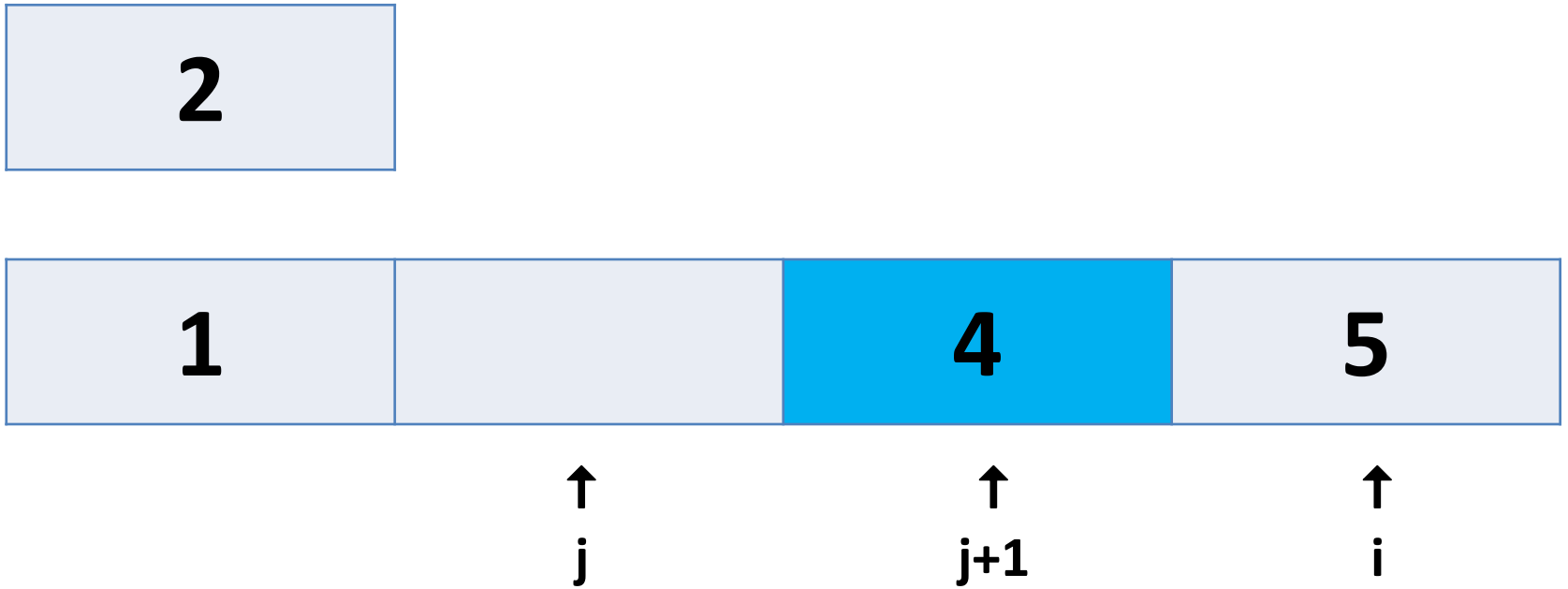
Insertion Sort

- Compare
- Store=



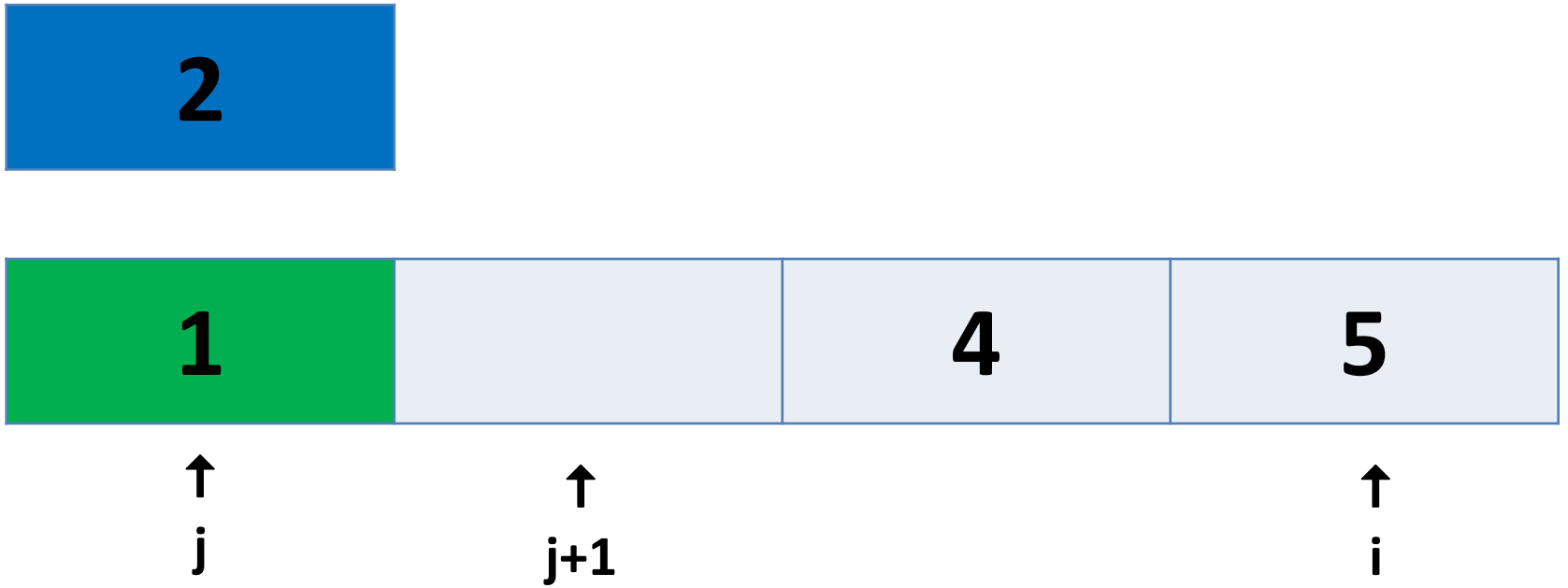
Insertion Sort

- Move
- Store=



Insertion Sort

- Compare
- Store=



Insertion Sort

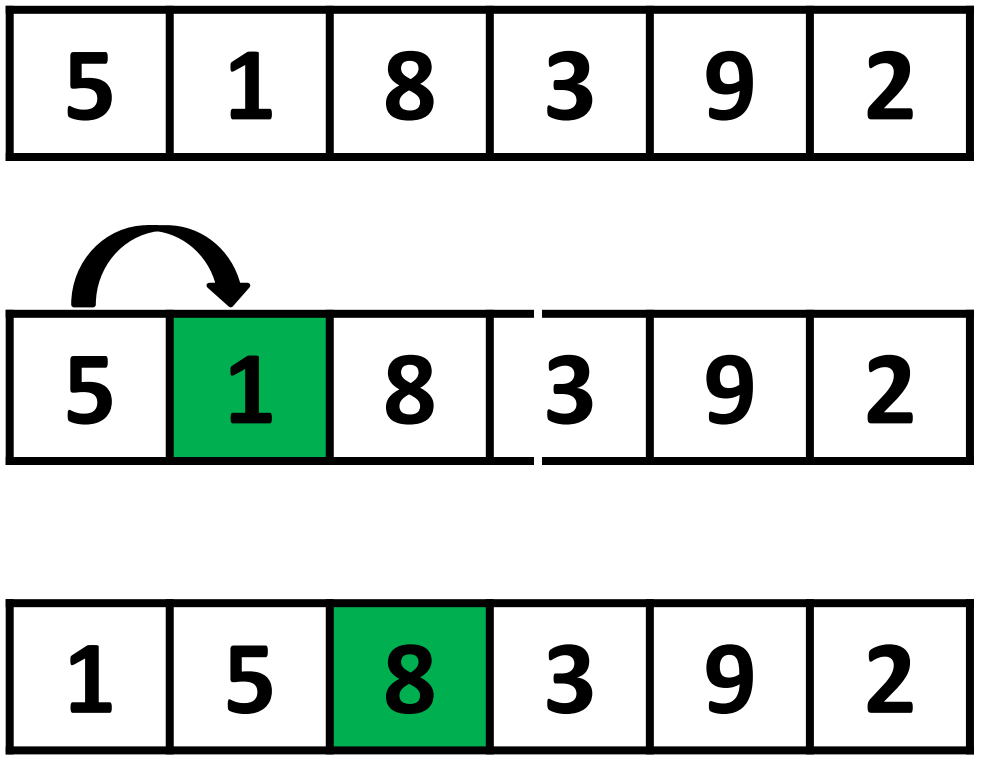
- Compare
- Store=

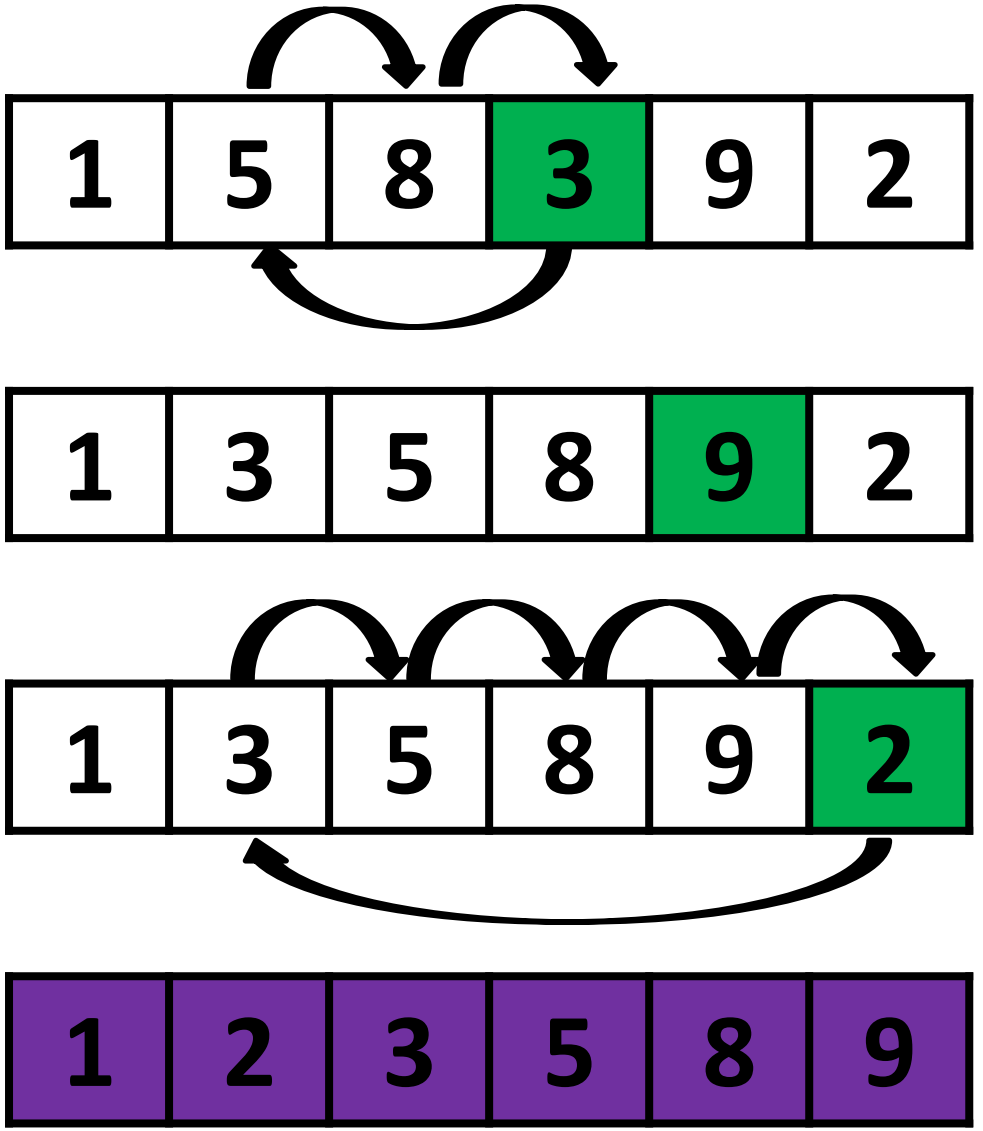
Insertion Sort

- Array is now sorted

1	2	4	5

❑ Example 2:





Insertion Sort

- What is the output of insertion sort after the 1st iteration given the following sequence of numbers: **7 3 5 1 9 8 4 6**

  a) 3 7 5 1 9 8 4 6

  b) 1 3 7 5 9 8 4 6

  c) 3 4 1 5 6 8 7 9

  d) 1 3 4 5 6 7 8 9

Insertion Sort

- What is the output of insertion sort after the 1st iteration given the following sequence of numbers: **7 3 5 1 9 8 4 6**

  a) 3 7 5 1 9 8 4 6

  b) 1 3 7 5 9 8 4 6

  c) 3 4 1 5 6 8 7 9

  d) 1 3 4 5 6 7 8 9

Insertion Sort

- What is the output of insertion sort after the 2nd iteration given the following sequence of numbers: **7 3 5 1 9 8 4 6**

  a) 3 5 7 1 9 8 4 6

  b) 1 3 7 5 9 8 4 6

  c) 3 4 1 5 6 8 7 9

  d) 1 3 4 5 6 7 8 9

Insertion Sort

- What is the output of insertion sort after the 2nd iteration given the following sequence of numbers: **7 3 5 1 9 8 4 6**

  a) 3 5 7 1 9 8 4 6

  b) 1 3 7 5 9 8 4 6

  c) 3 4 1 5 6 8 7 9

  d) 1 3 4 5 6 7 8 9

Insertion Sort

- **Time Complexity:** $O(n^2)$

- Example of worst case

5	4	3	2	1

THANKS FOR YOUR TIME