Sort

By

Dr. Tisha

Courtesy: Dr. Myers

What is a Sort?

4 8 9 2 6 3 5 1 7

Sort these numbers in ascending order

Descending order

Sort() in java

Java uses **Timsort** — a hybrid of **Merge Sort** and **Insertion Sort**.

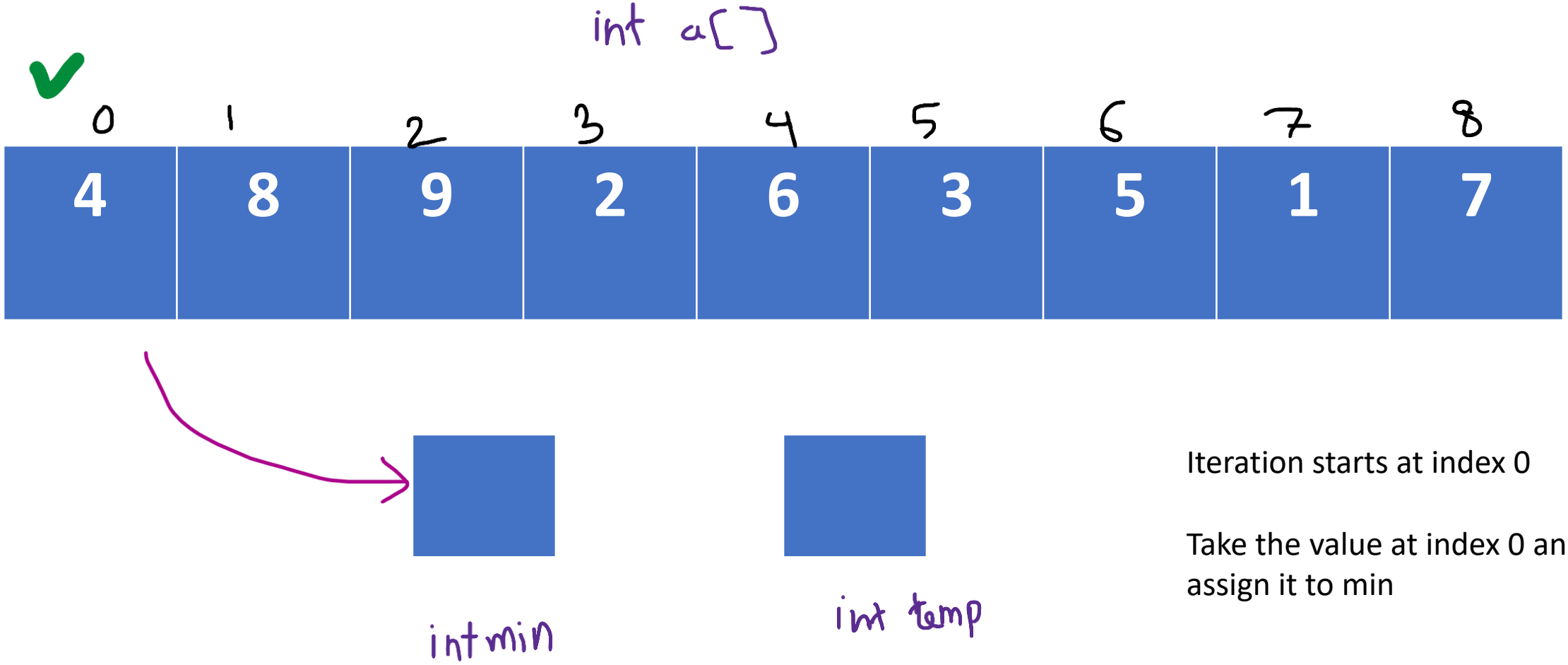
Some sorting Algorithms

- Selection sort
- Insertion sort
- Merge sort
- Quick sort
- Bubble sort

etc.

If we can do it directly just calling the sort() why we need to learn all these?

Selection Sort

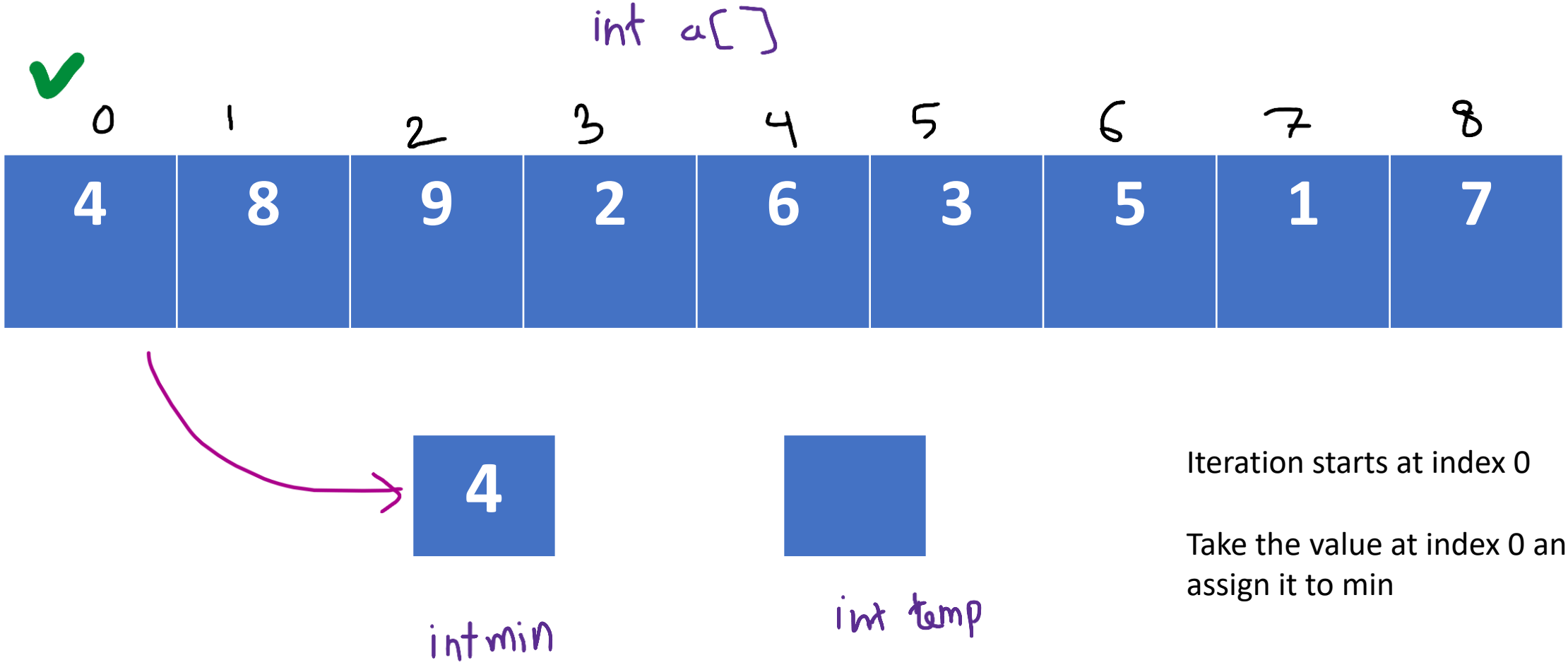


Iteration starts at index 0

Take the value at index 0 and assign it to min

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

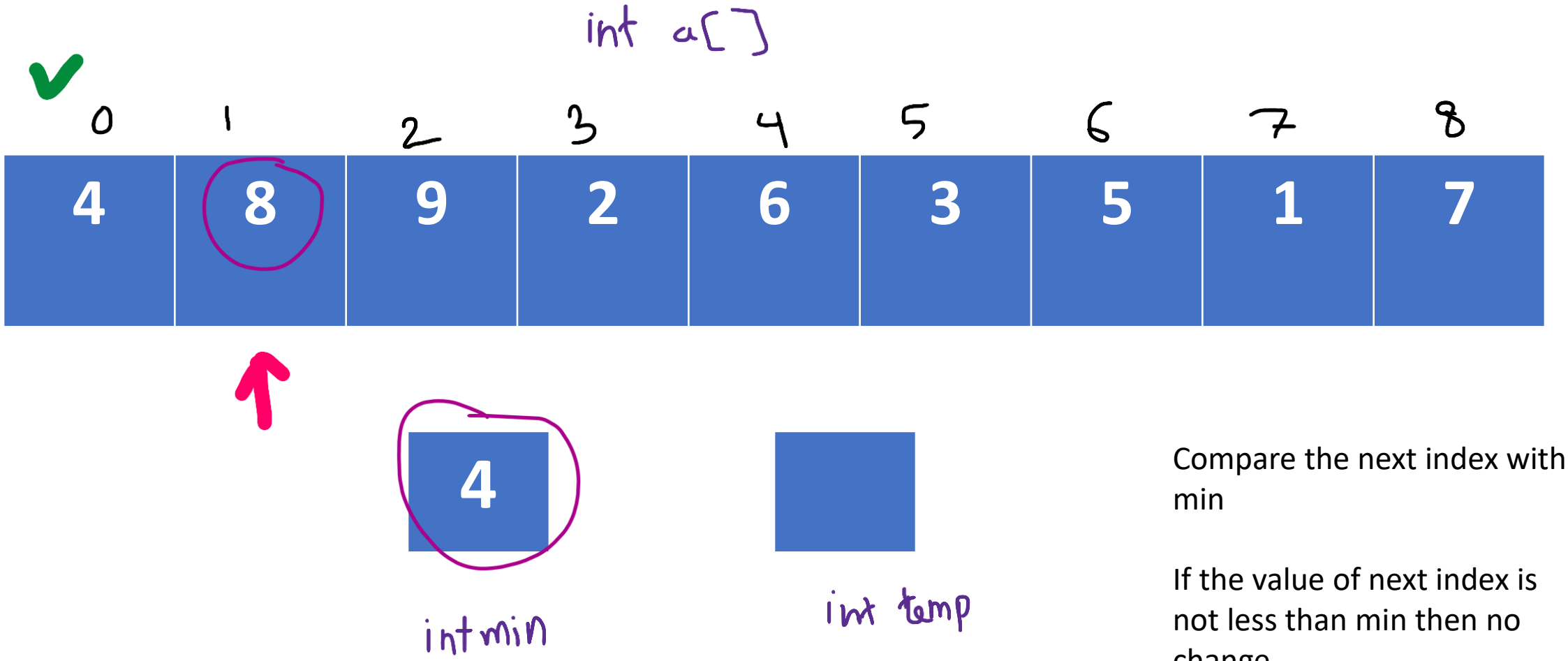


Iteration starts at index 0

Take the value at index 0 and assign it to min

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

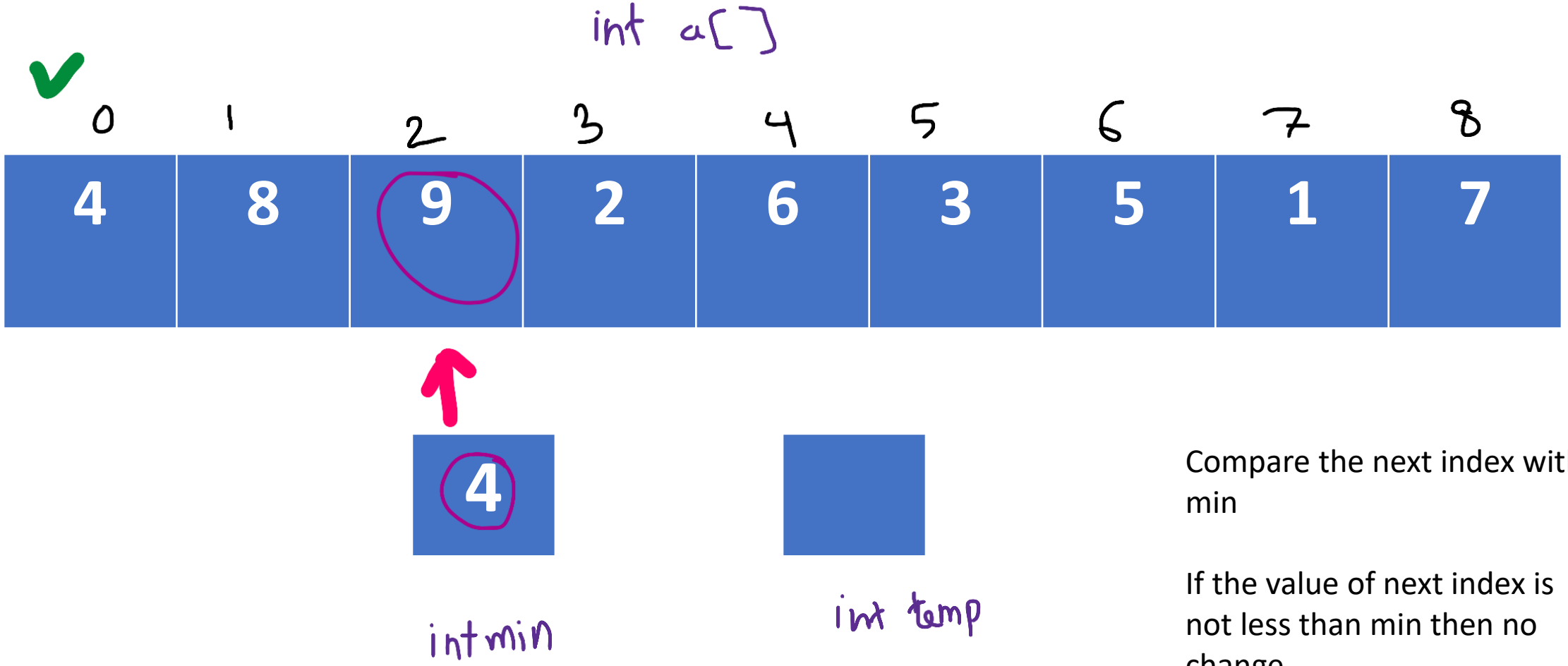


Compare the next index with min

If the value of next index is not less than min then no change

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

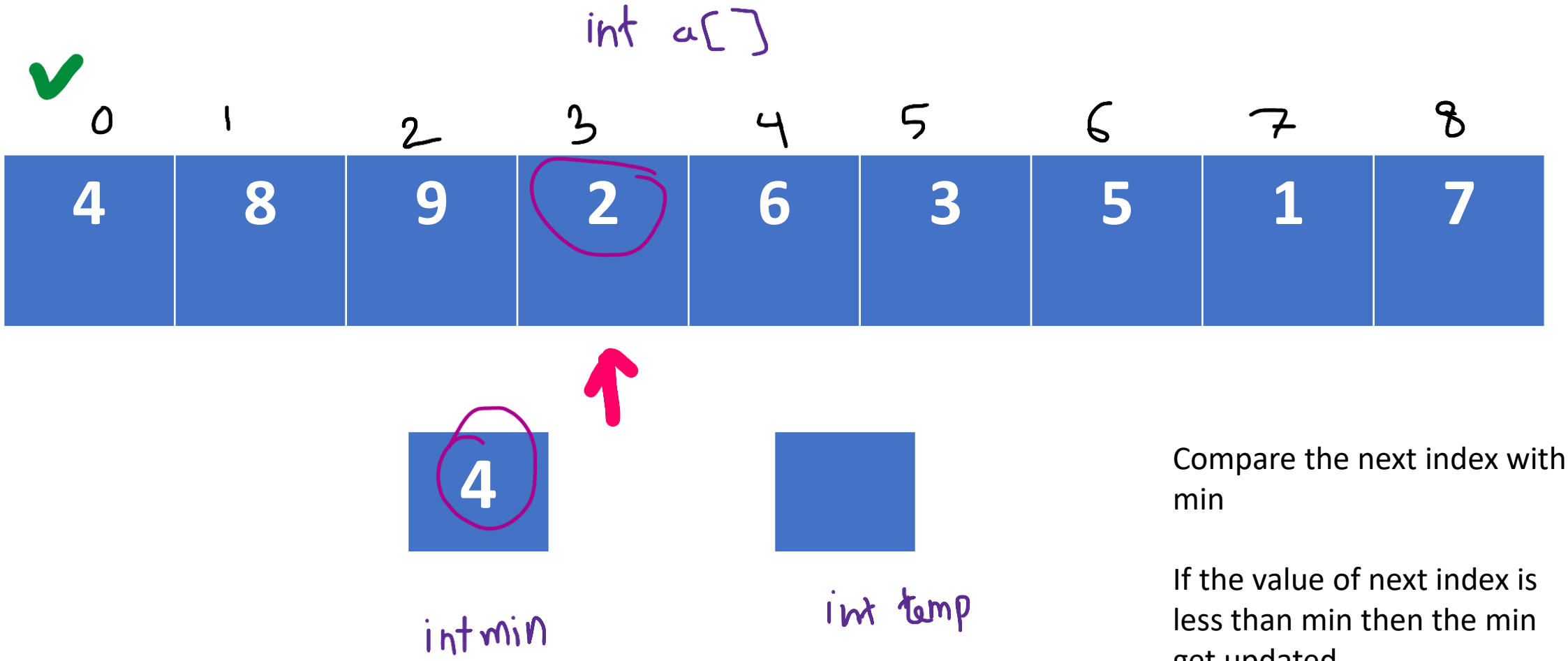


Compare the next index with min

If the value of next index is not less than min then no change

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

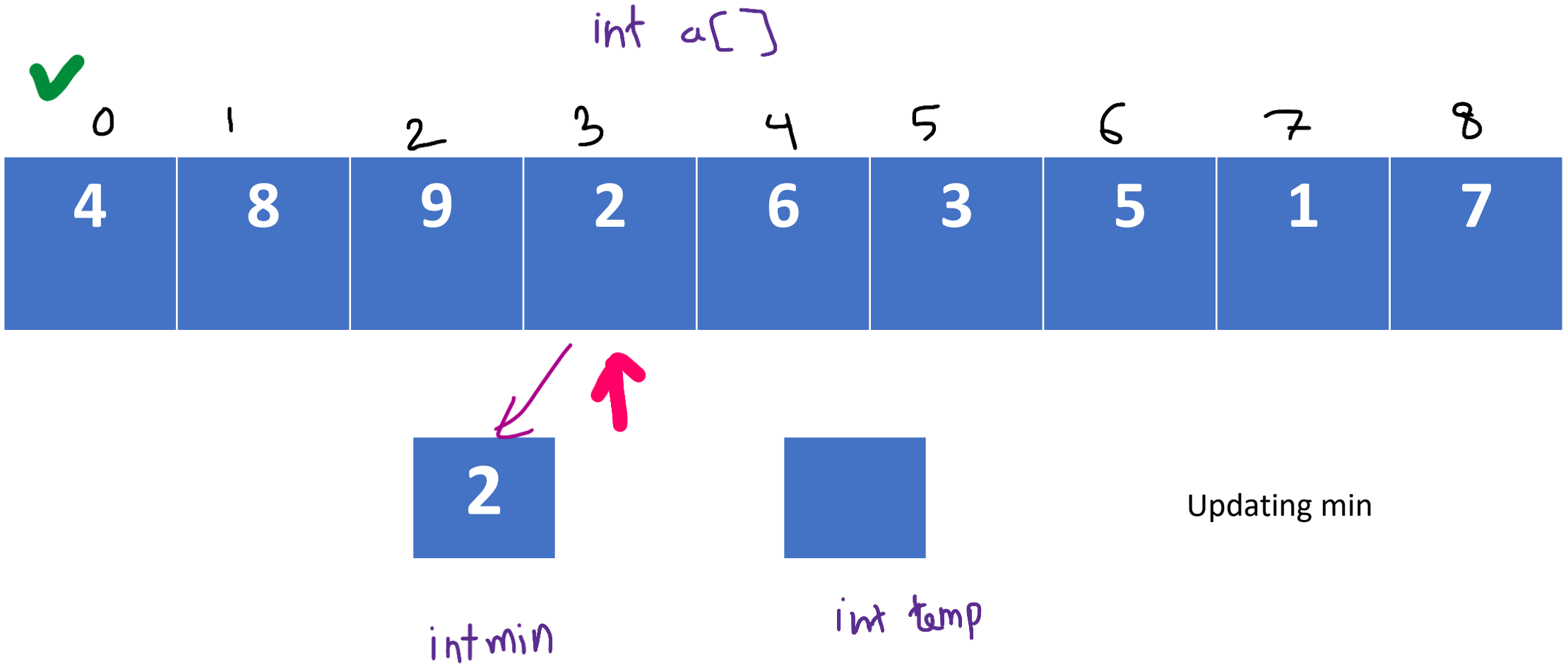


Compare the next index with min

If the value of next index is less than min then the min get updated.

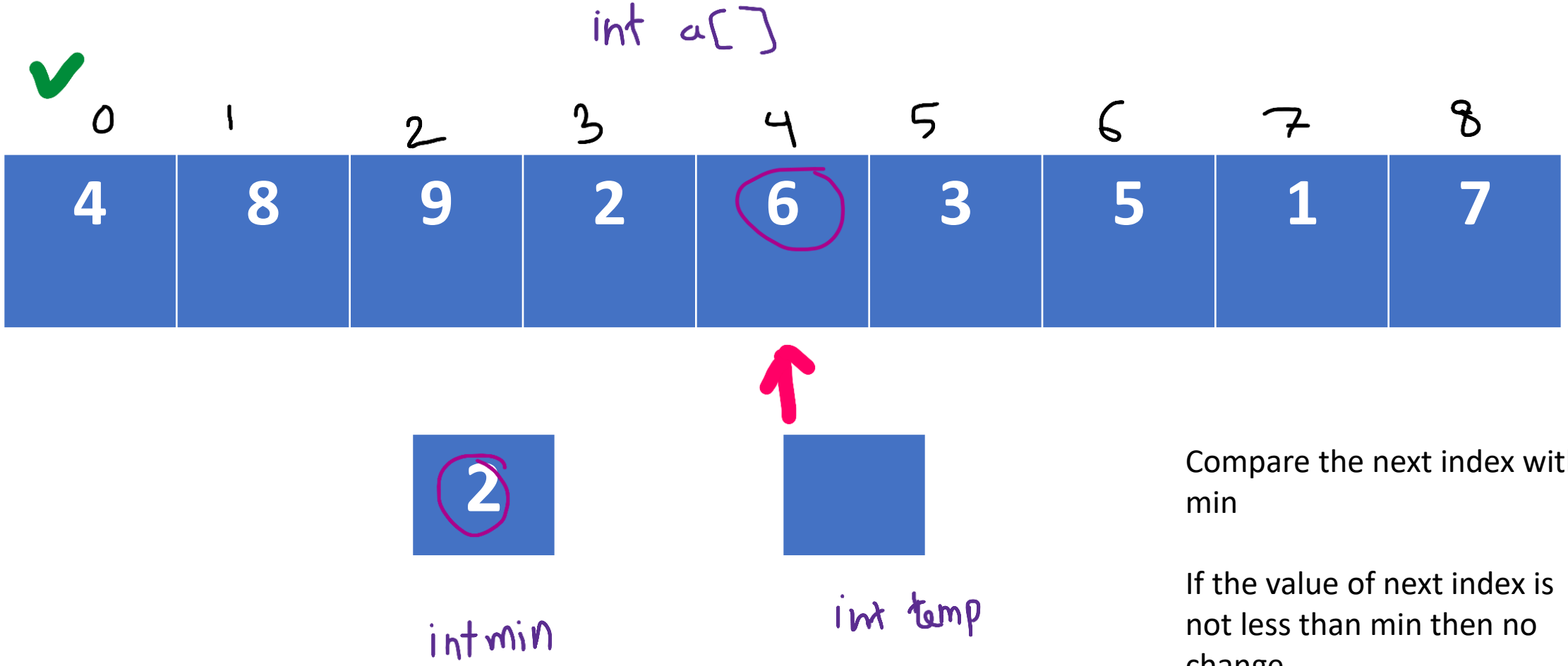
https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

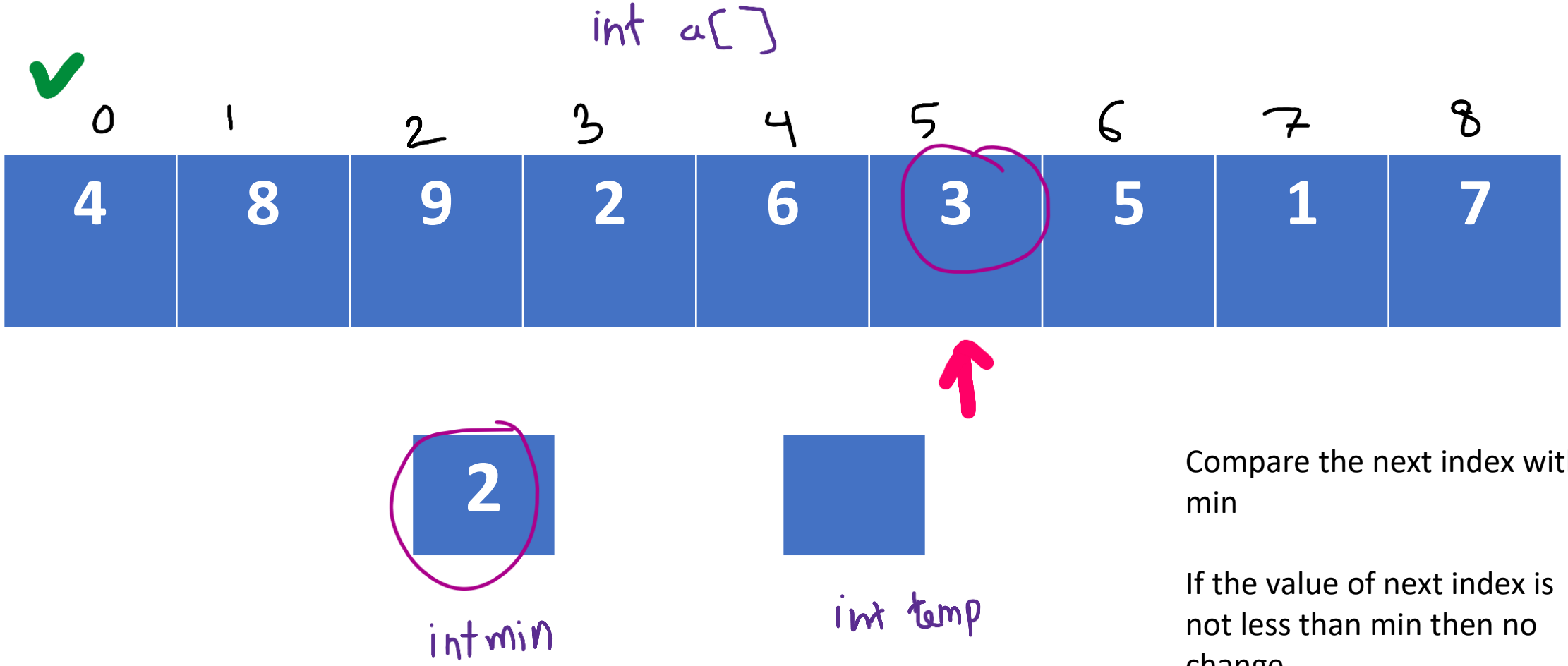


Compare the next index with min

If the value of next index is not less than min then no change

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

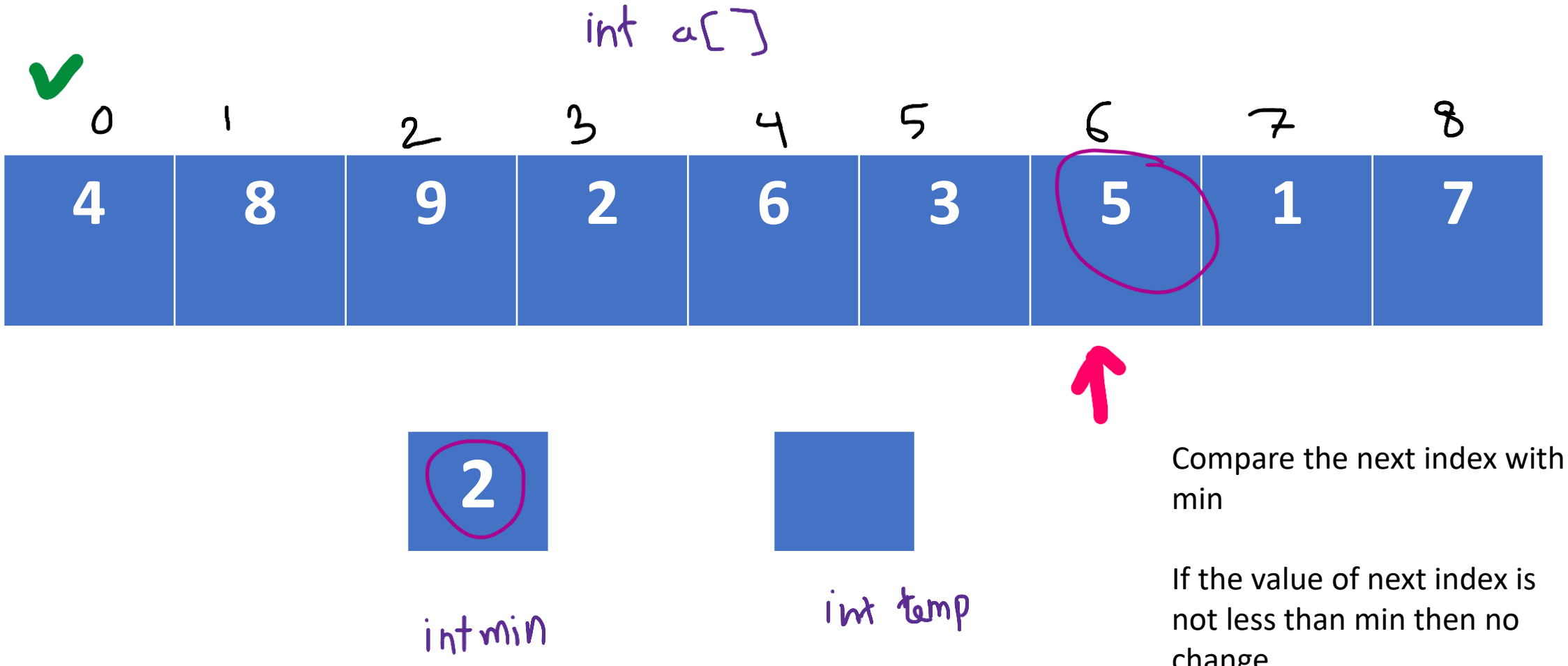


Compare the next index with min

If the value of next index is not less than min then no change

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

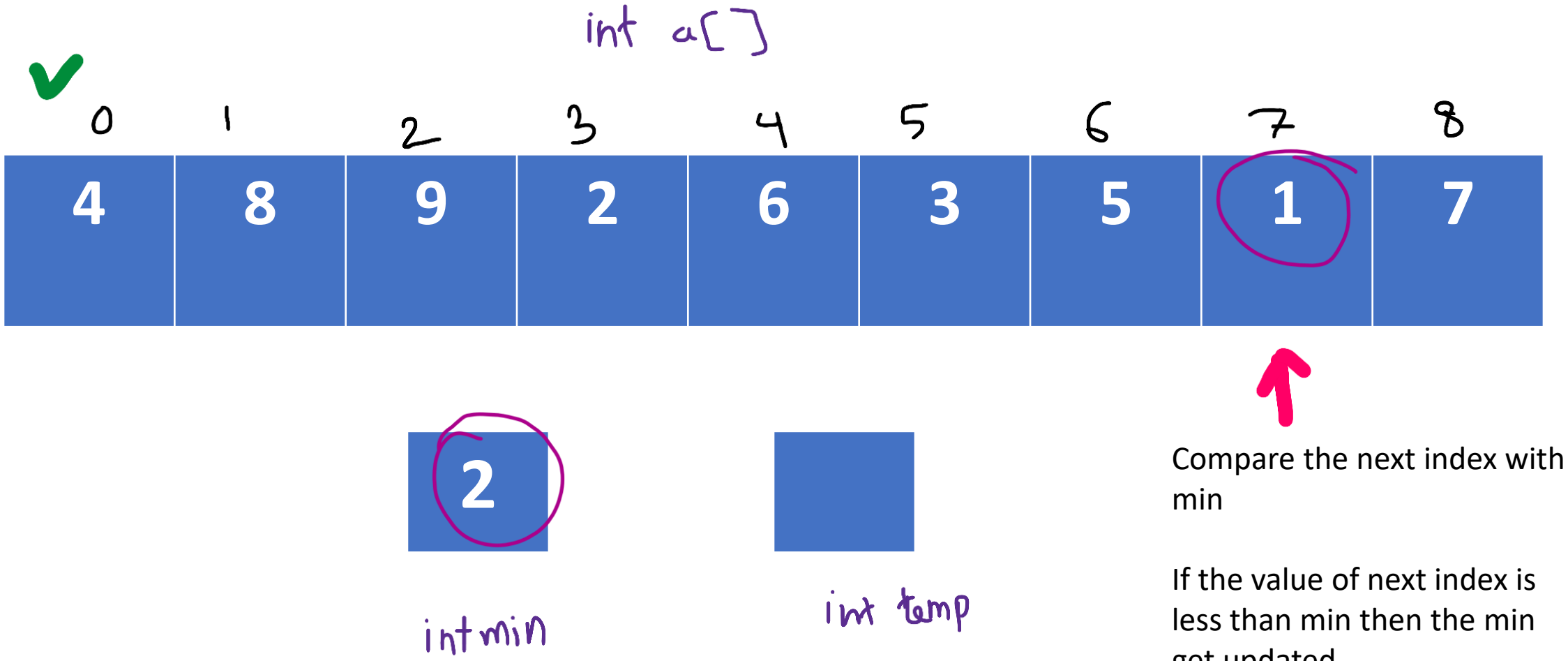


Compare the next index with min

If the value of next index is not less than min then no change

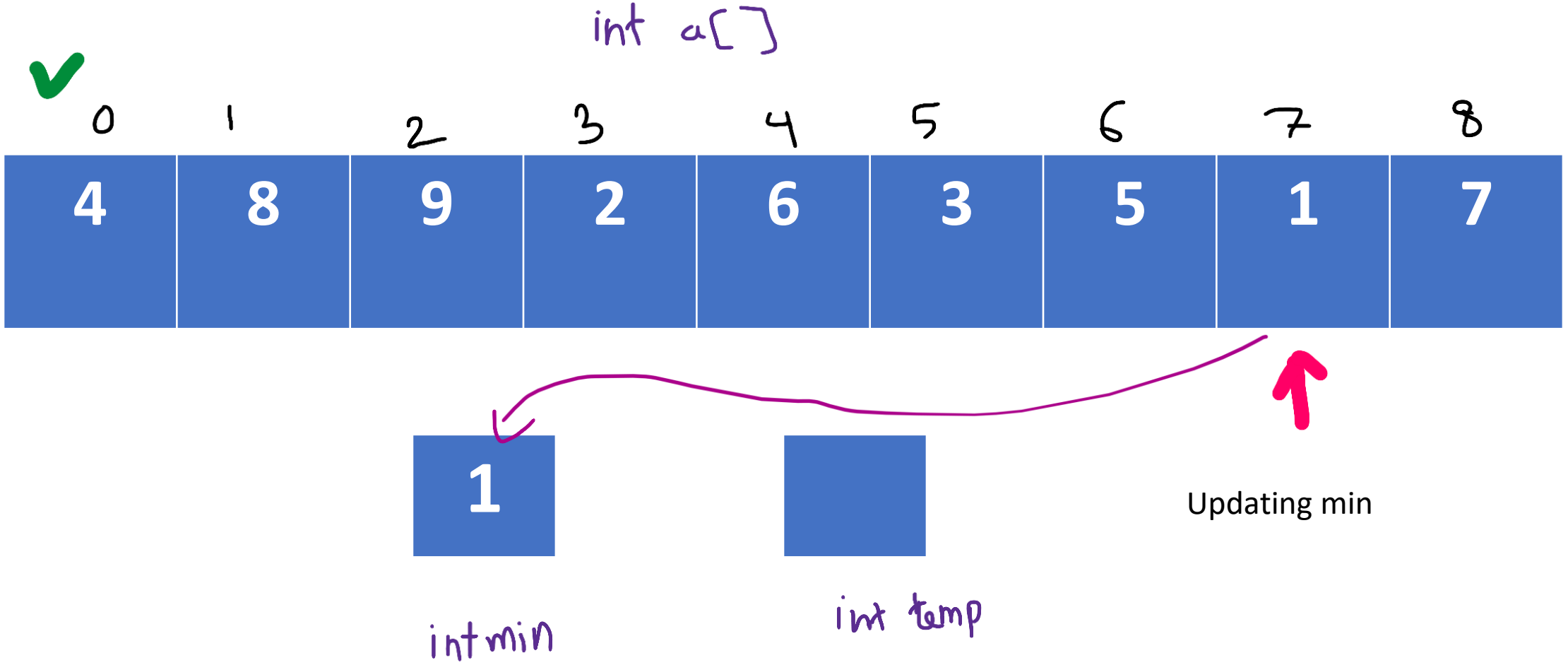
https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



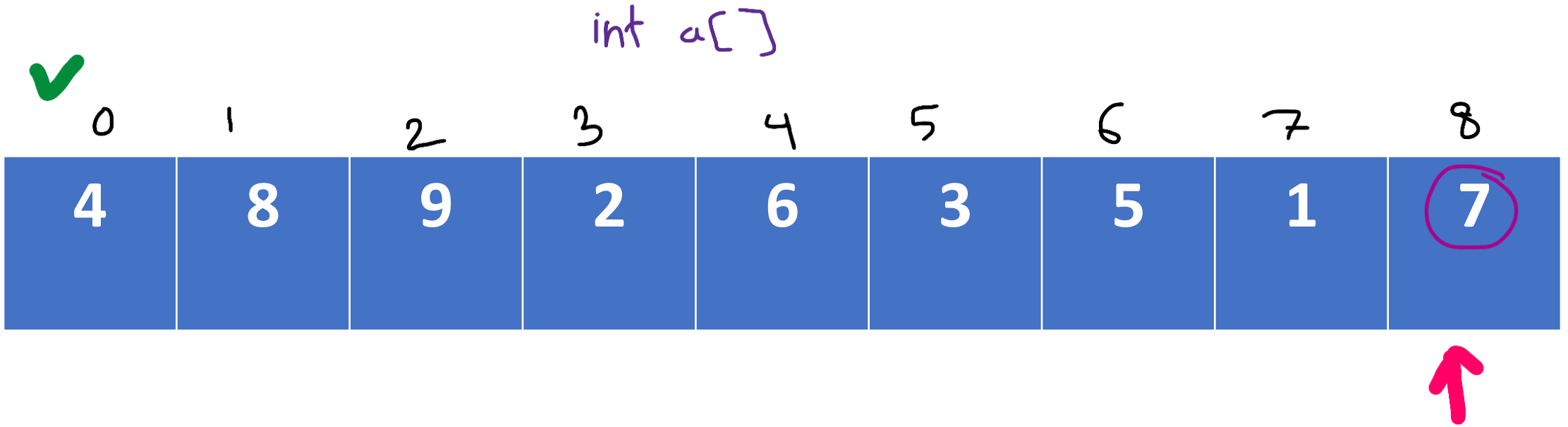
https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

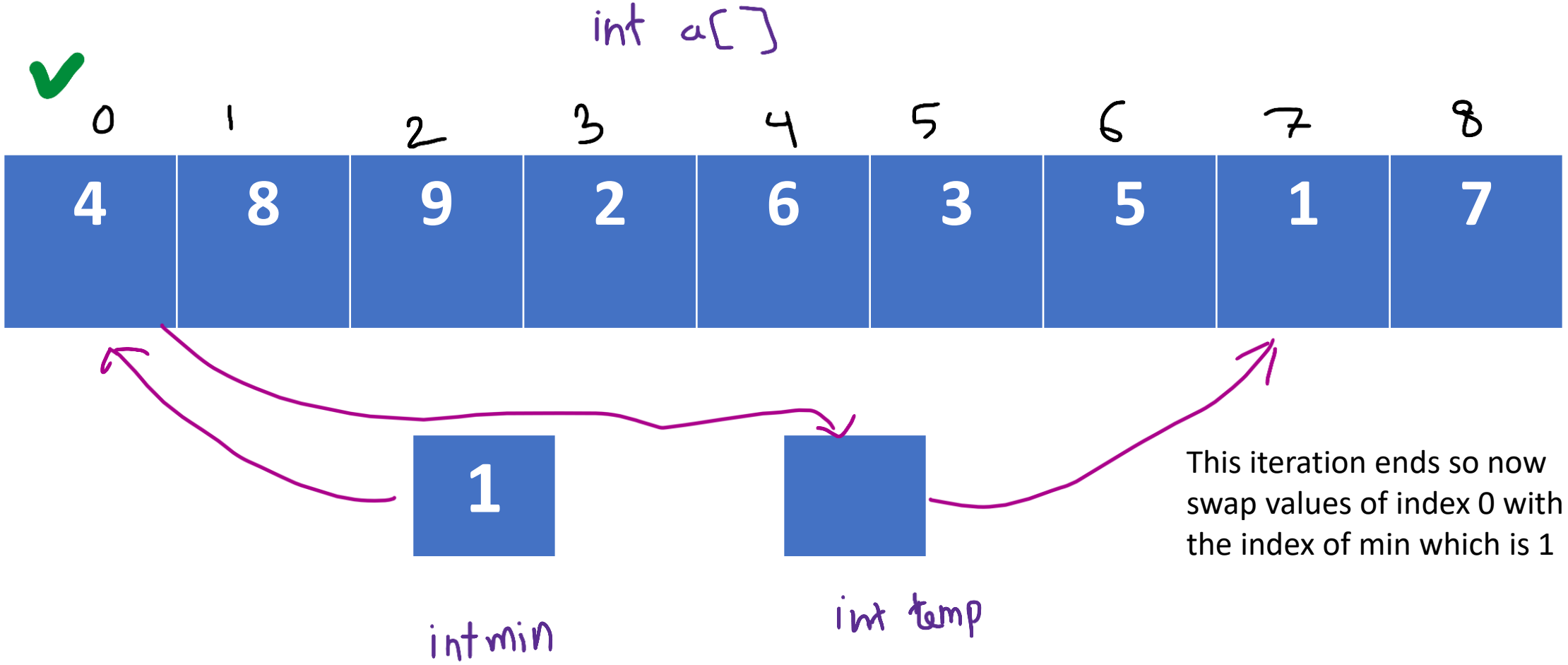


Compare the next index with min

If the value of next index is not less than min then no change

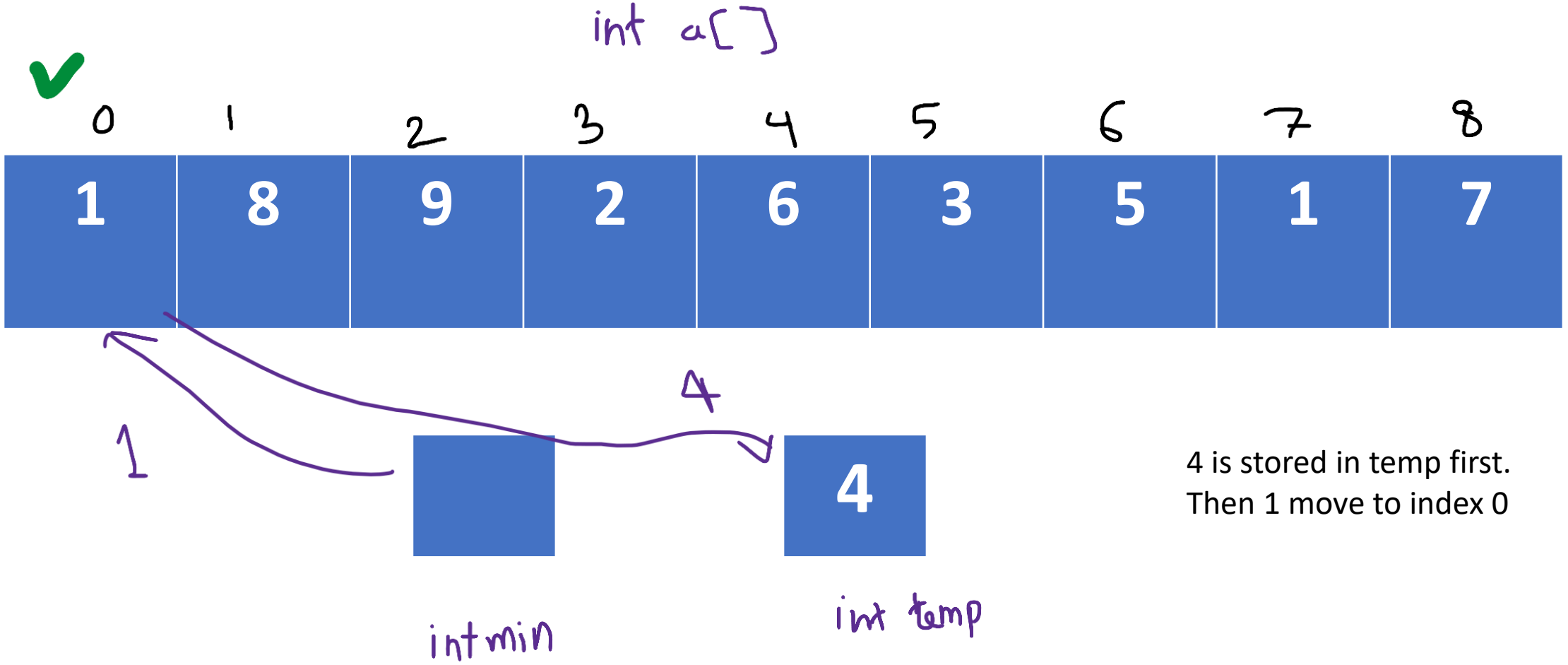
https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



https://www.youtube.com/watch?v=EwjnF7rFLns

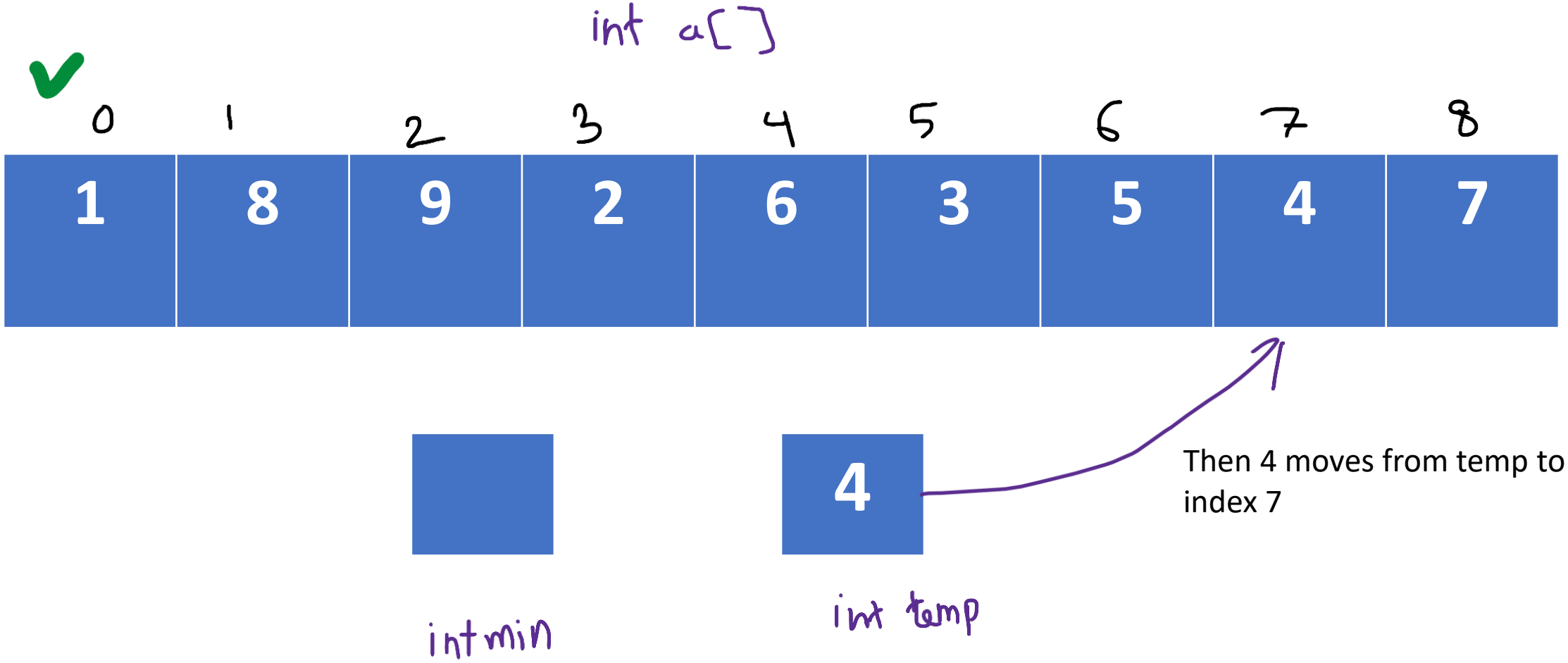
Selection Sort



4 is stored in temp first.
Then 1 move to index 0

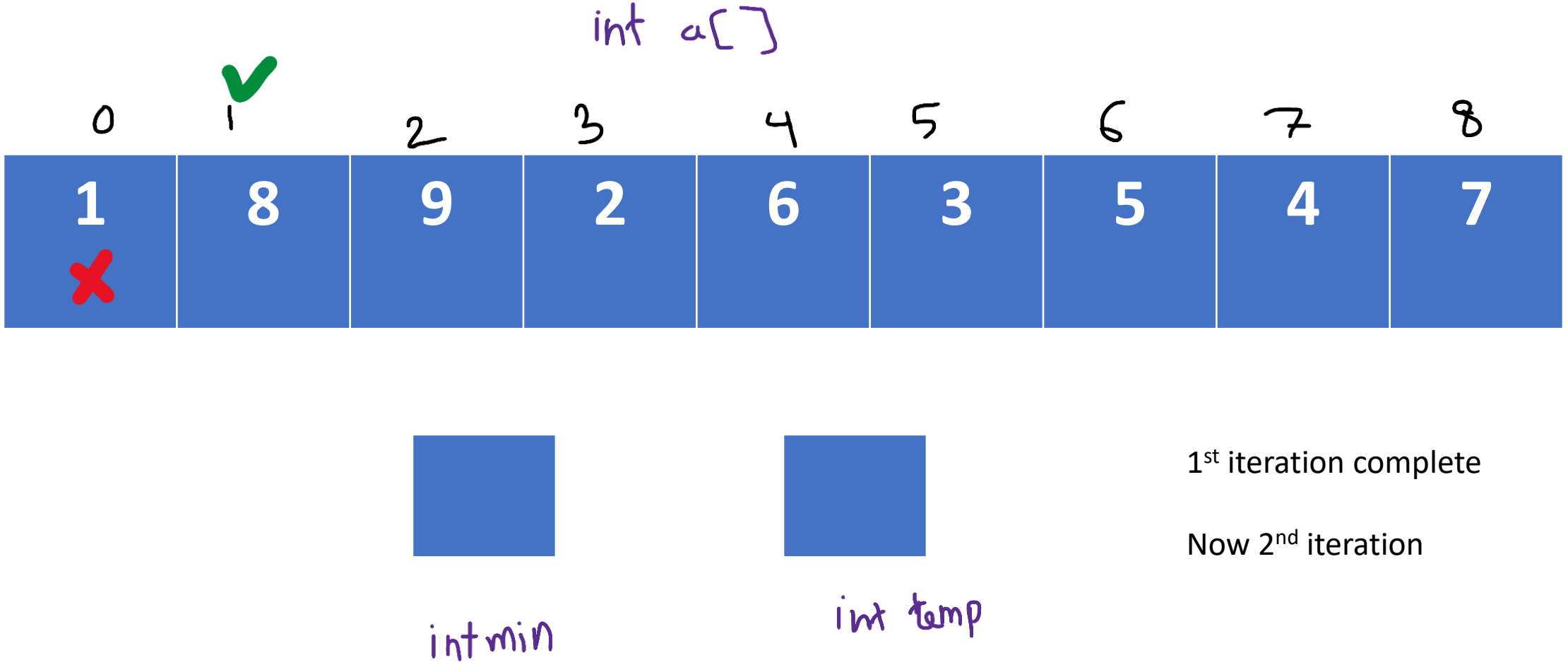
https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort

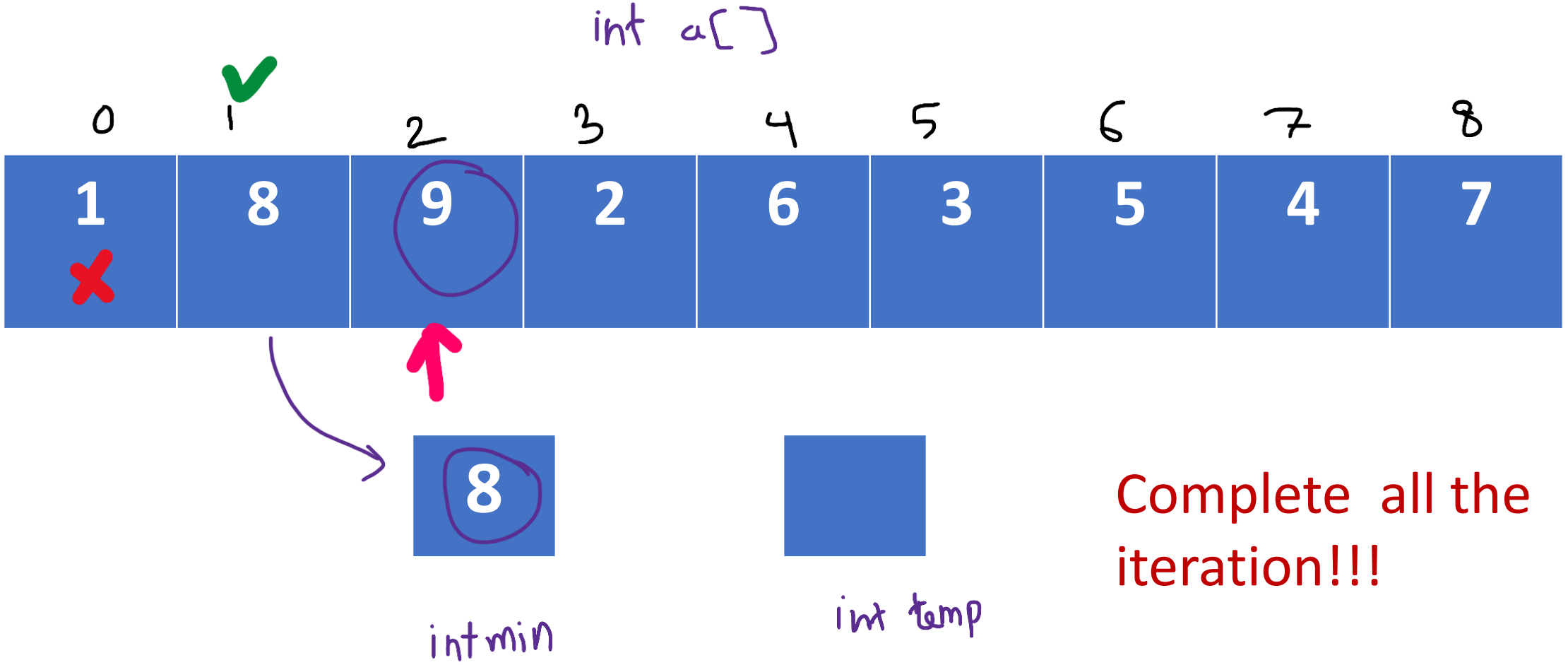


1st iteration complete

Now 2nd iteration

https://www.youtube.com/watch?v=EwjnF7rFLns

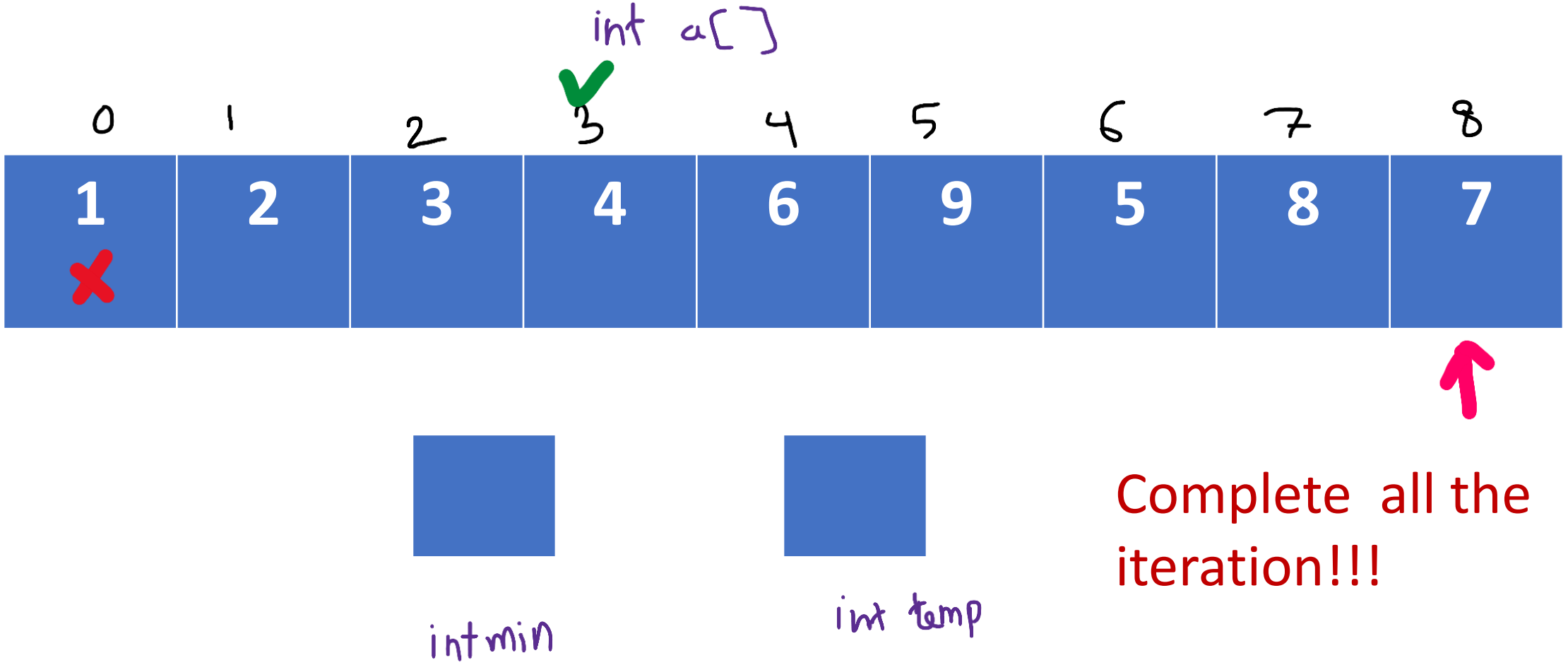
Selection Sort



Complete all the iteration!!!

https://www.youtube.com/watch?v=EwjnF7rFLns

Selection Sort



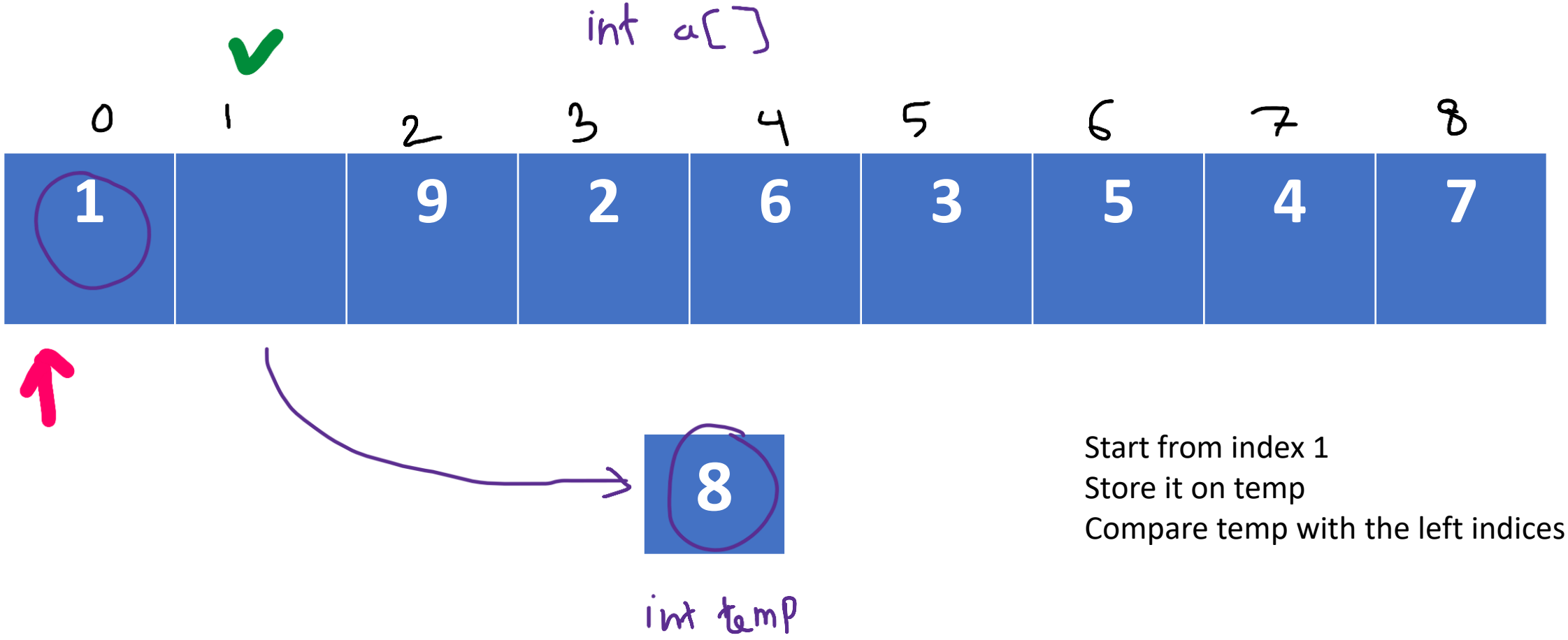
https://www.youtube.com/watch?v=EwjnF7rFLns

Practice Que:

Make this code work !
What would be the O()?
Print in the opposite order.

```java
public static void selectionSort(int[] a) {
  // Outer loop: find the smallest remaining element
  // and swap into position i
  for (int i = 0; i < a.length - 1; i++) {

    int minValue = a[i];
    int minIndex = i;

    // Inner loop: traverse the remaining elements and
    // identify the minimum
    for (int j = i + 1; j < a.length; j++) {
      if (a[j] < minValue) {
        minValue = a[j];
        minIndex = j;
      }
    }

    // Swap positions i and minIndex
    int temp = a[i];
    a[i] = a[minIndex];
    a[minIndex] = temp;
  }

}
```

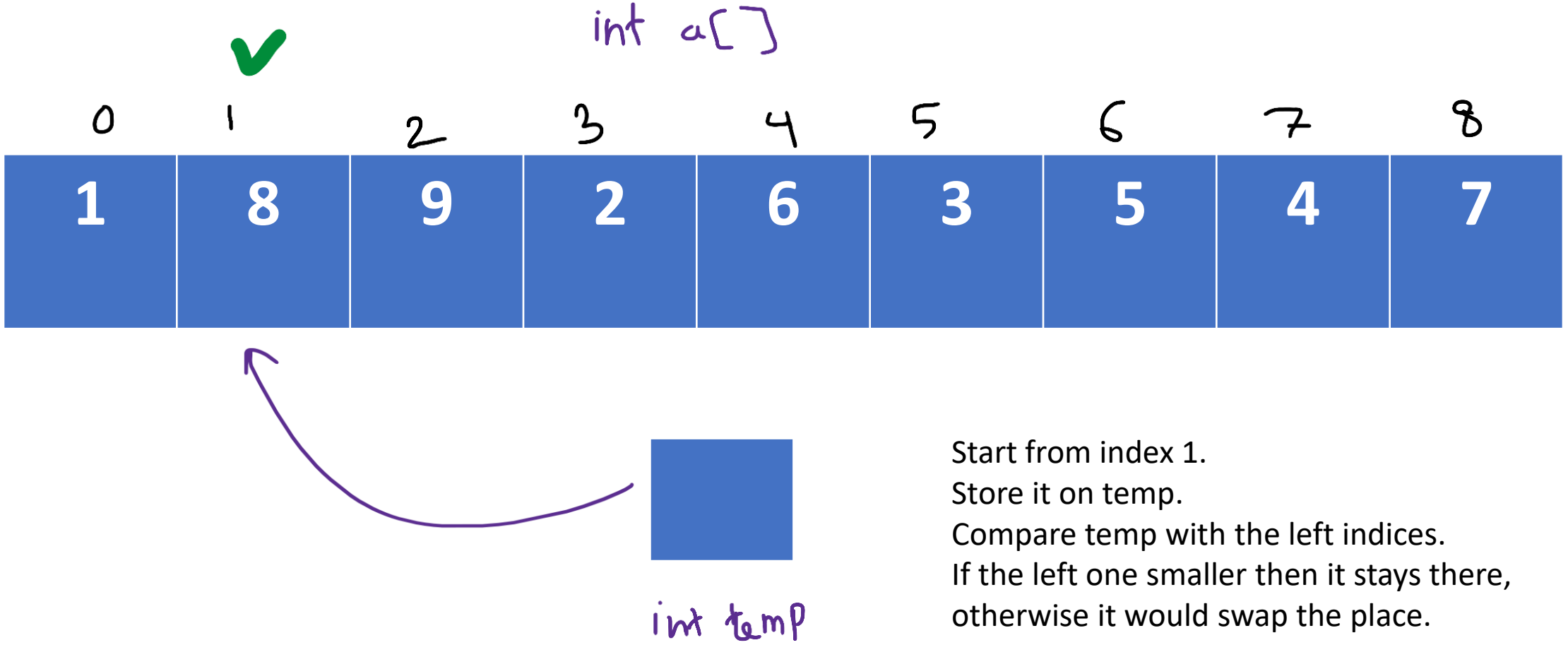
Insertion Sort



Start from index 1
Store it on temp
Compare temp with the left indices

https://www.youtube.com/watch?v=8mJ-OhcfpYg

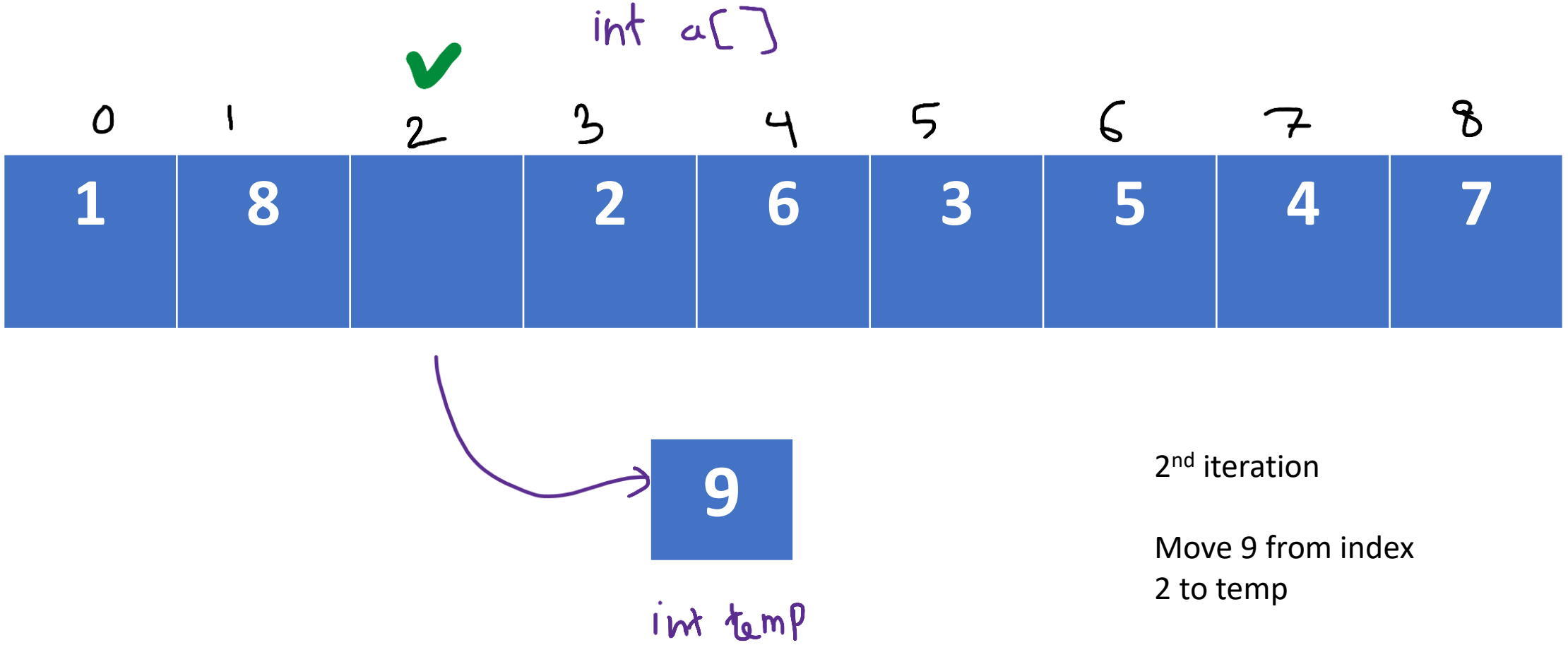
Insertion Sort



Start from index 1.
Store it on temp.
Compare temp with the left indices.
If the left one smaller then it stays there,
otherwise it would swap the place.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort

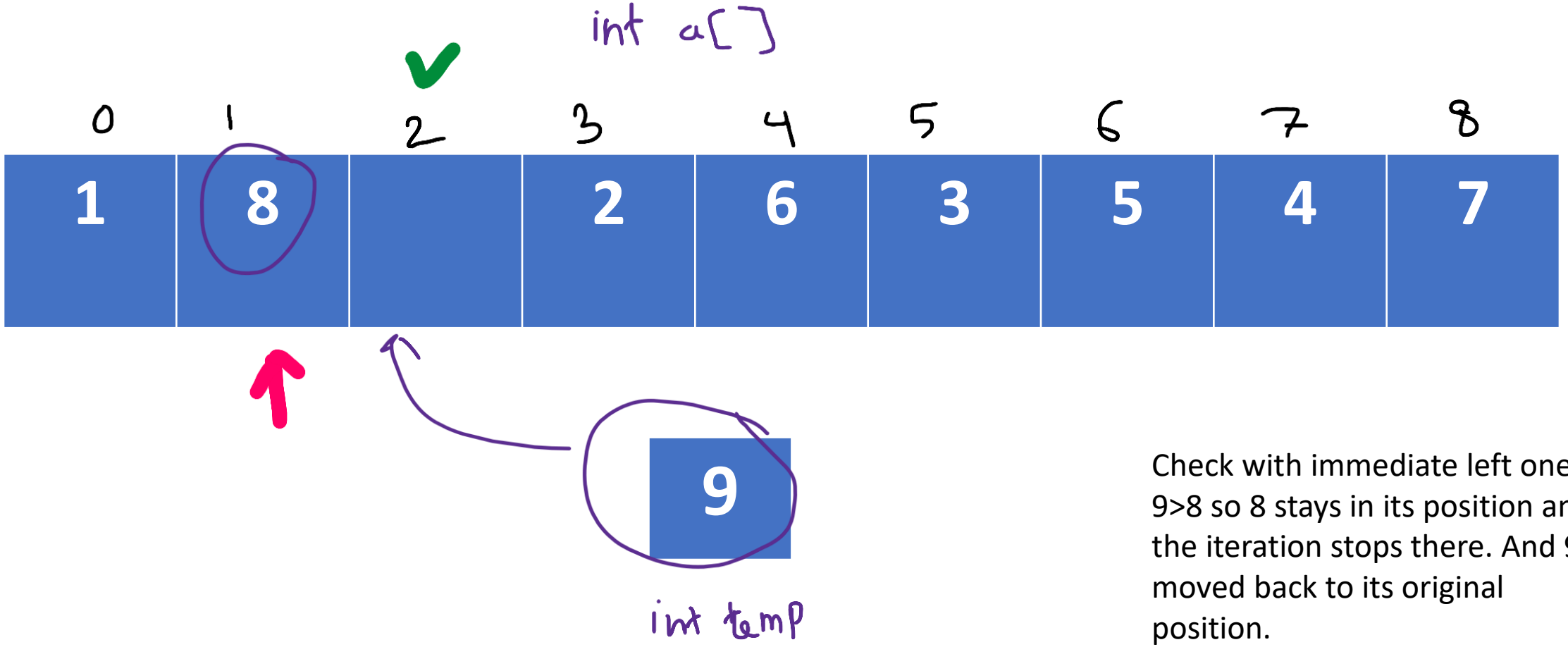


2nd iteration

Move 9 from index 2 to temp

https://www.youtube.com/watch?v=8mJ-OhcfpYg

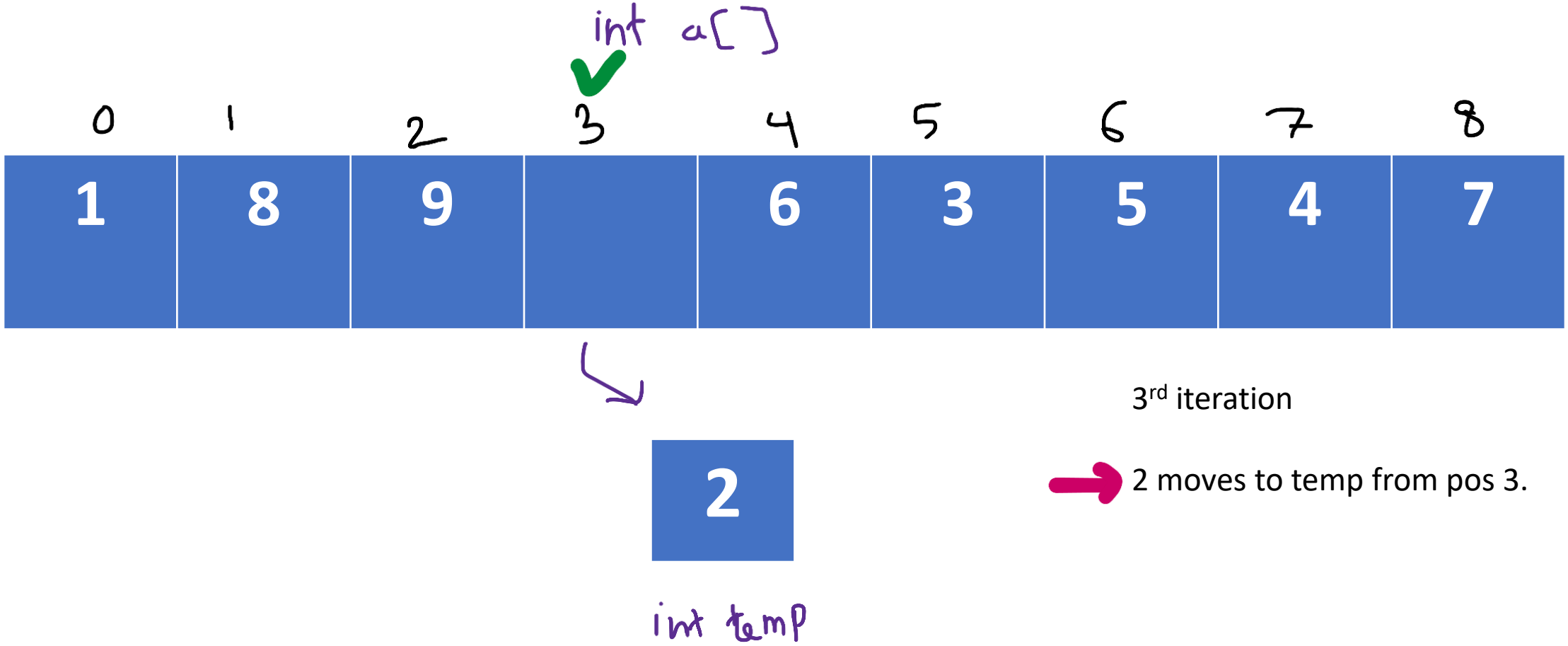
Insertion Sort



Check with immediate left one. 9>8 so 8 stays in its position and the iteration stops there. And 9 moved back to its original position.

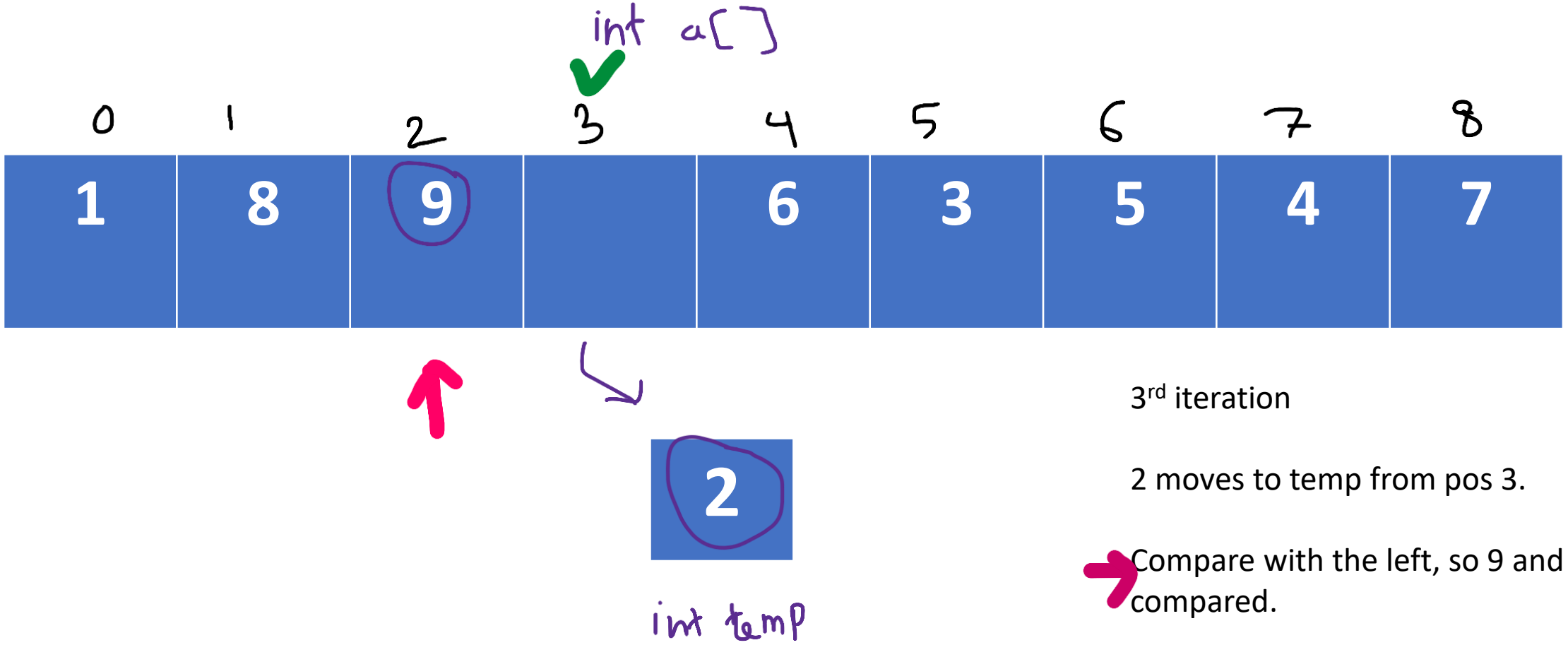
https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



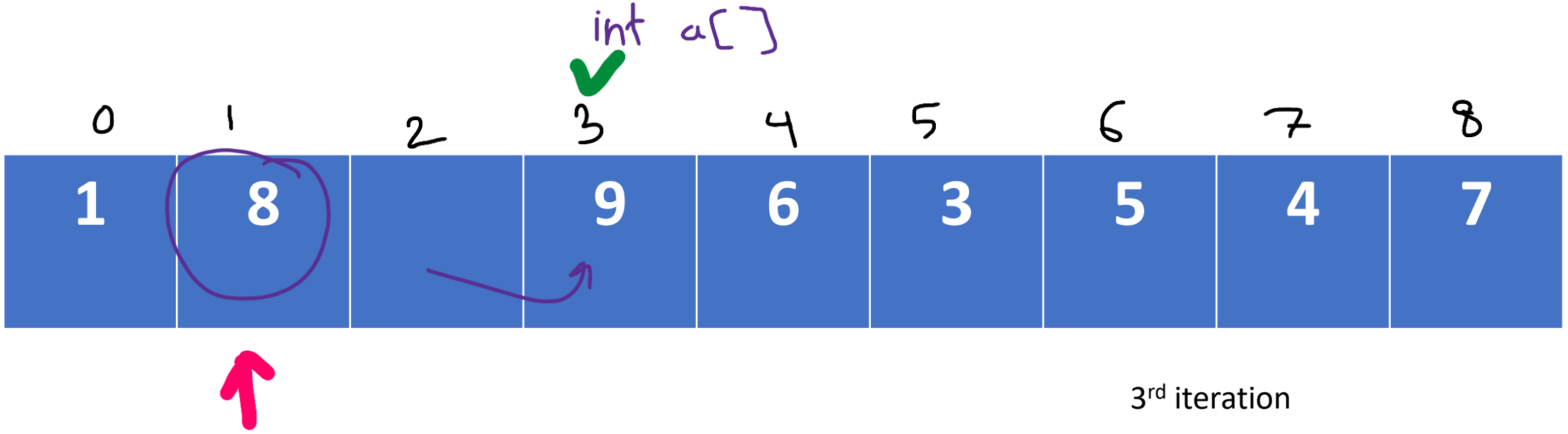
3rd iteration

2 moves to temp from pos 3.

Compare with the left, so 9 and 2 compared.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

2 moves to temp from pos 3.

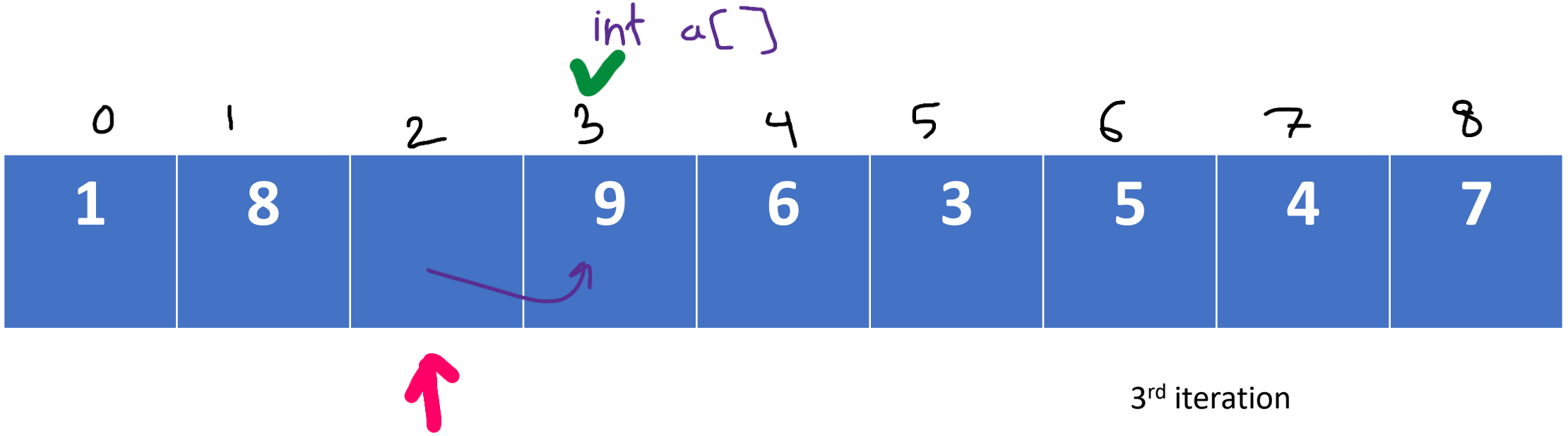
Compare with the left, so 9 and 2 compared.

9>2 so 9 moved to right.

Now 8 and 2 get compared.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

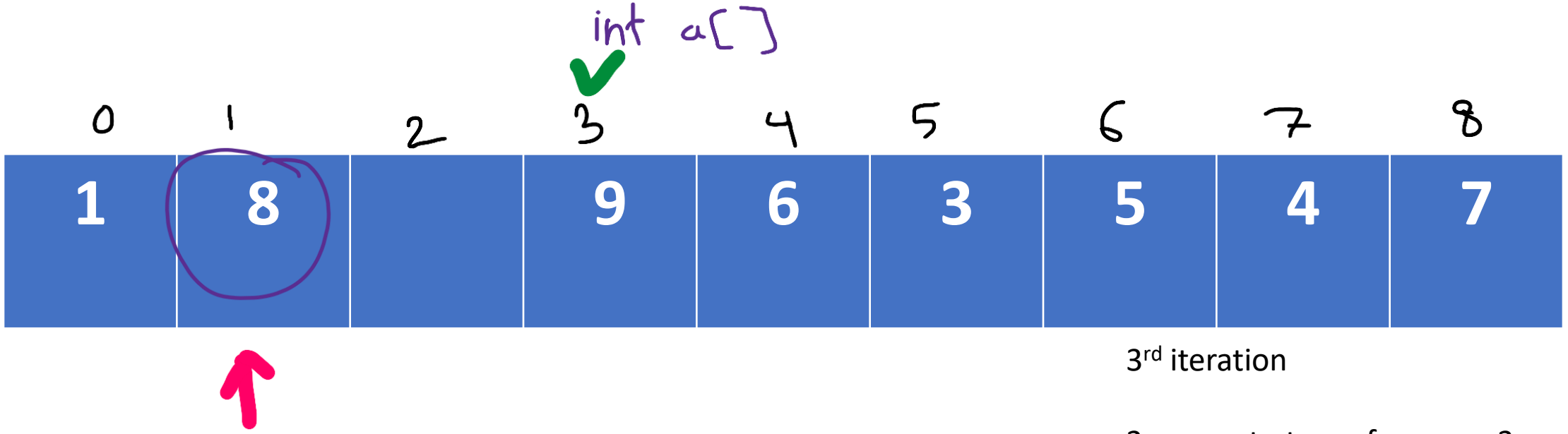
2 moves to temp from pos 3.

Compare with the left, so 9 and 2 compared.

9>2 so 9 moved to right.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

2 moves to temp from pos 3.

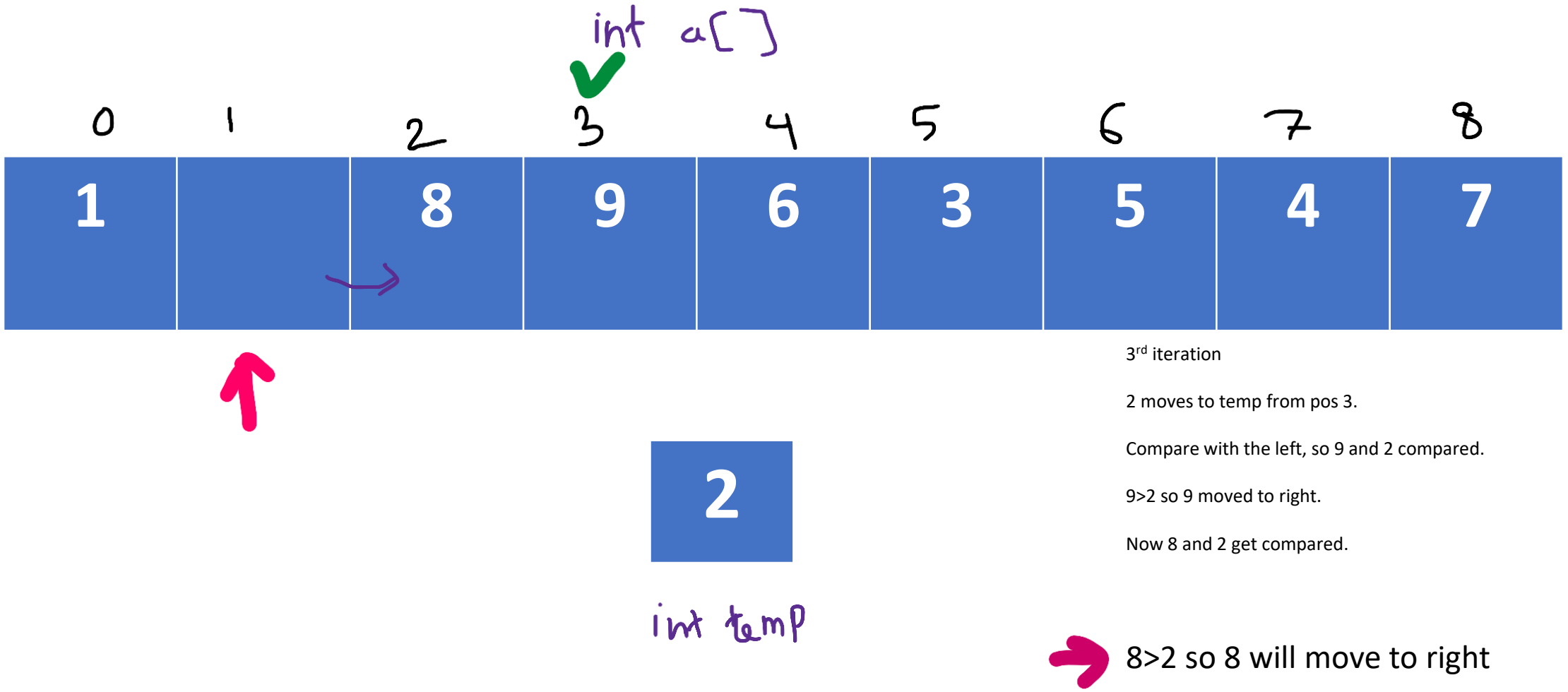
Compare with the left, so 9 and 2 compared.

9>2 so 9 moved to right.

Now 8 and 2 get compared.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

2 moves to temp from pos 3.

Compare with the left, so 9 and 2 compared.

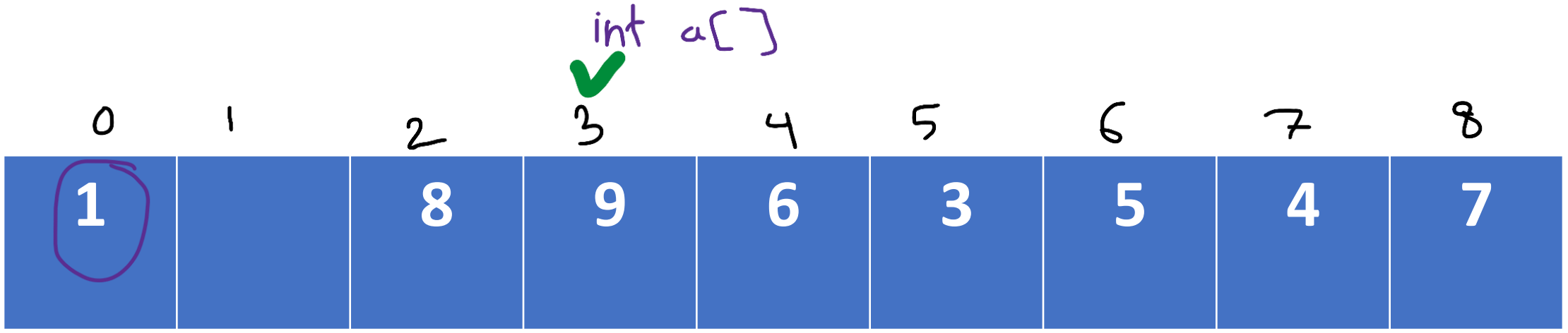
9>2 so 9 moved to right.

Now 8 and 2 get compared.

8>2 so 8 will move to right

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

2 moves to temp from pos 3.

Compare with the left, so 9 and 2 compared.

9>2 so 9 moved to right.

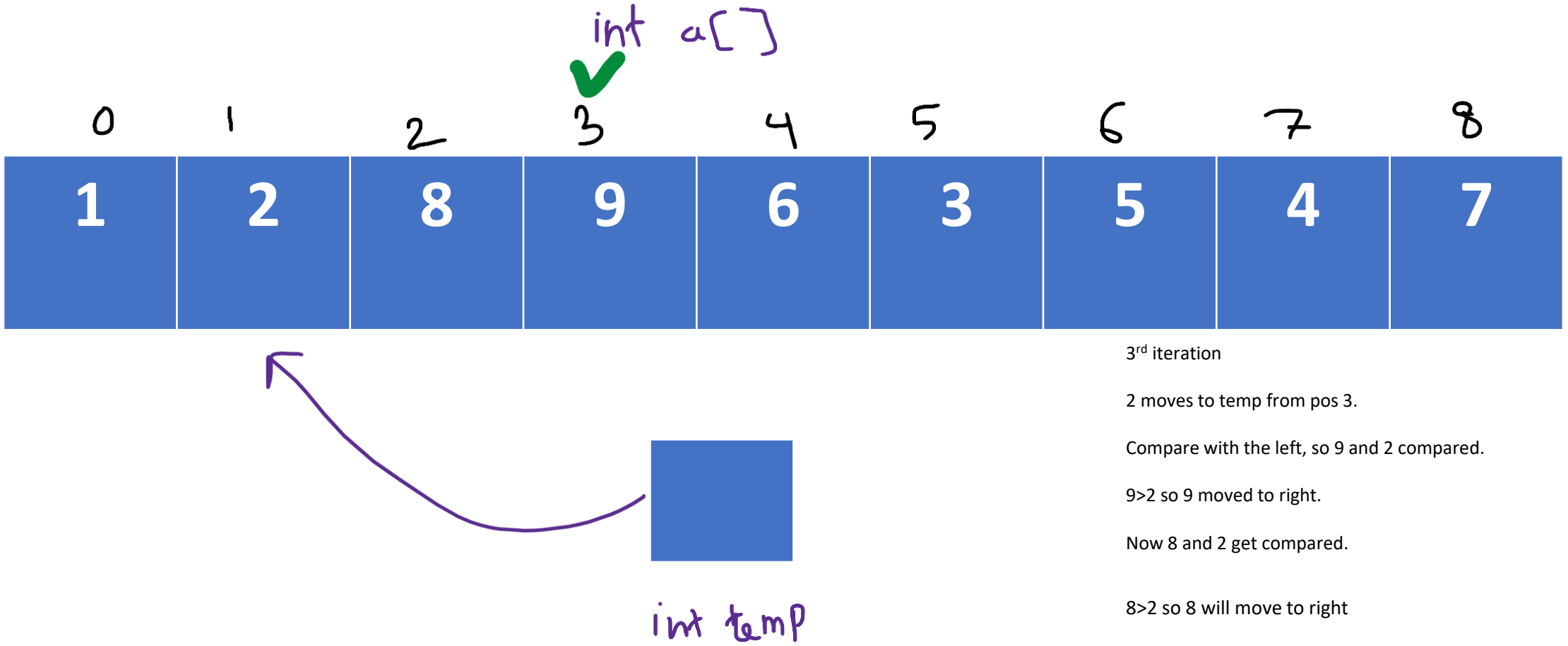
Now 8 and 2 get compared.

8>2 so 8 will move to right

Now 2 and 1 get compared.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



3rd iteration

2 moves to temp from pos 3.

Compare with the left, so 9 and 2 compared.

9>2 so 9 moved to right.

Now 8 and 2 get compared.

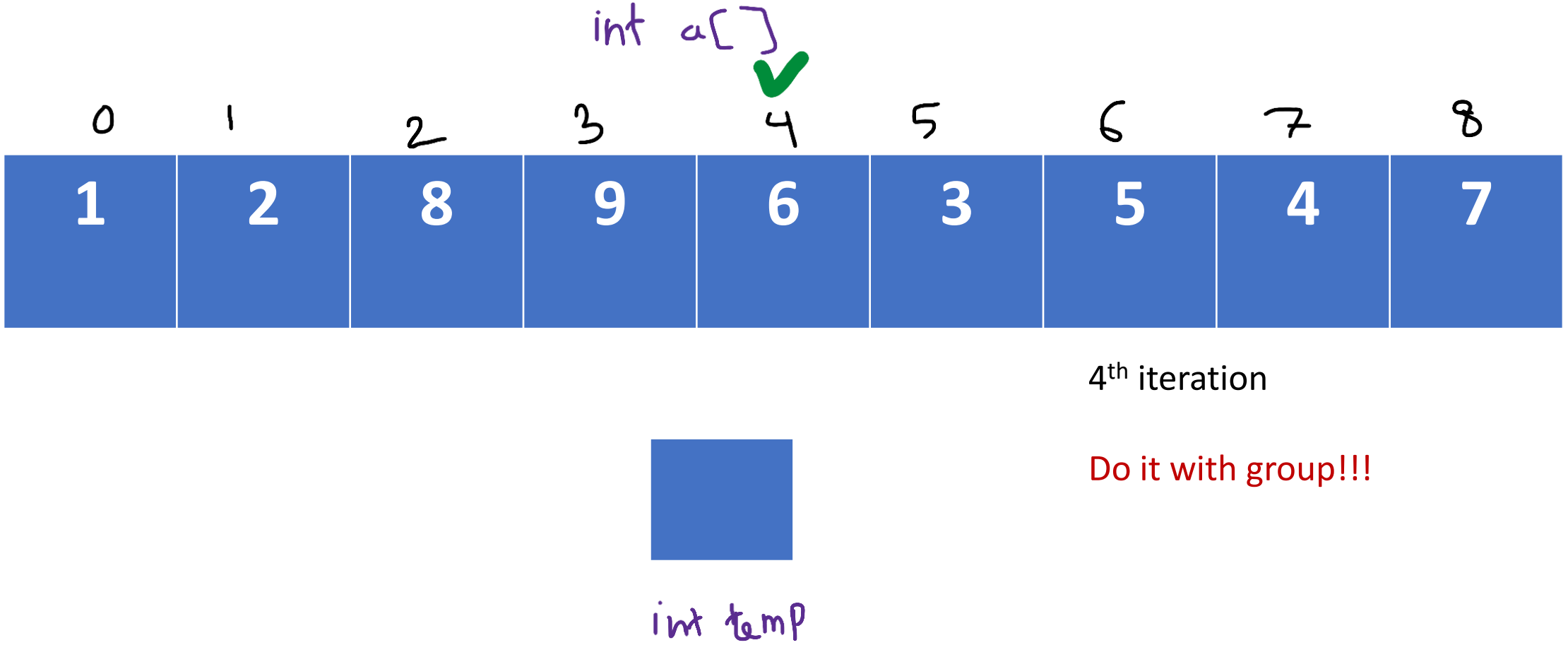
8>2 so 8 will move to right

Now 2 and 1 get compared.

As 1<2 so the iteration stops there and 2 moves to pos 1.

https://www.youtube.com/watch?v=8mJ-OhcfpYg

Insertion Sort



https://www.youtube.com/watch?v=8mJ-OhcfpYg

```
 * Insertion sort
 *
 * @param  a  the array to sort
 * @return  nothing, the array is sorted in-place
 */
public static void insertionSort(int[] a) {

  // Loop over the elements of the array
  for (int i = 1; i < a.length; i++) {
    // Swap item at position i into its correct position
    // relative to items 0 to i - 1
    int j = i;
    while (j > 0 && a[j - 1] > a[j]) {
      int temp = a[j - 1];
      a[j - 1] = a[j];
      a[j] = temp;
      j--;
    }
  }
}
```

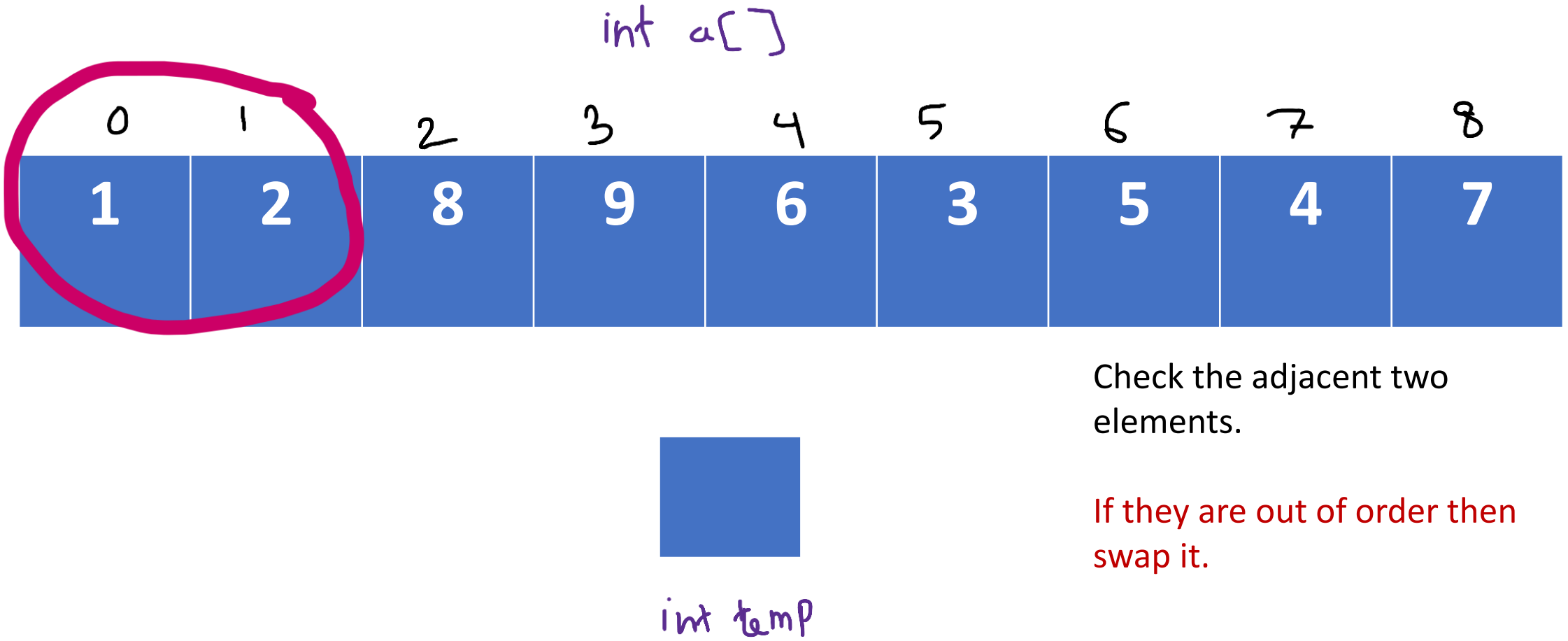
ClassPractice

- implement insertionSort function
- sort the input in ascending order
- sort in descending order
- What is O()? And why so?
- – O(n^2) because nested loop.

Bubble Sort

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

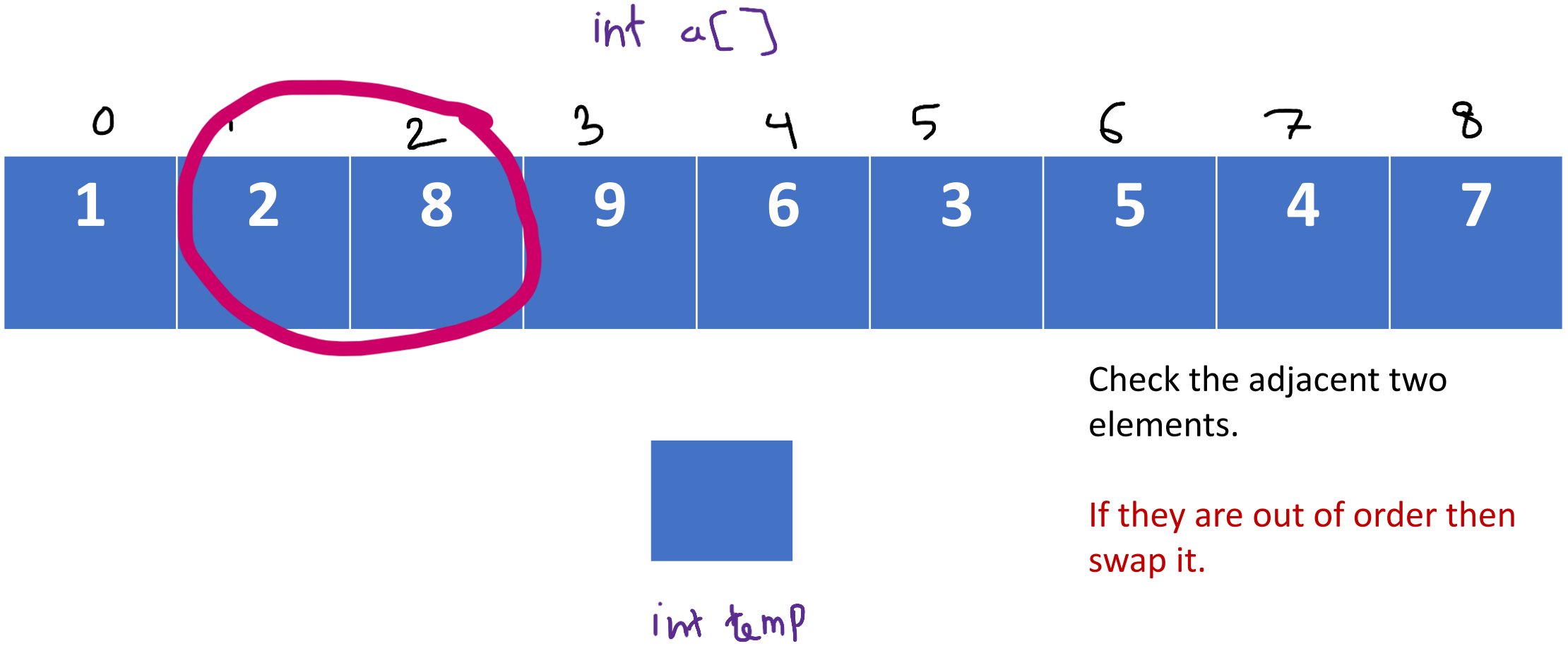


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

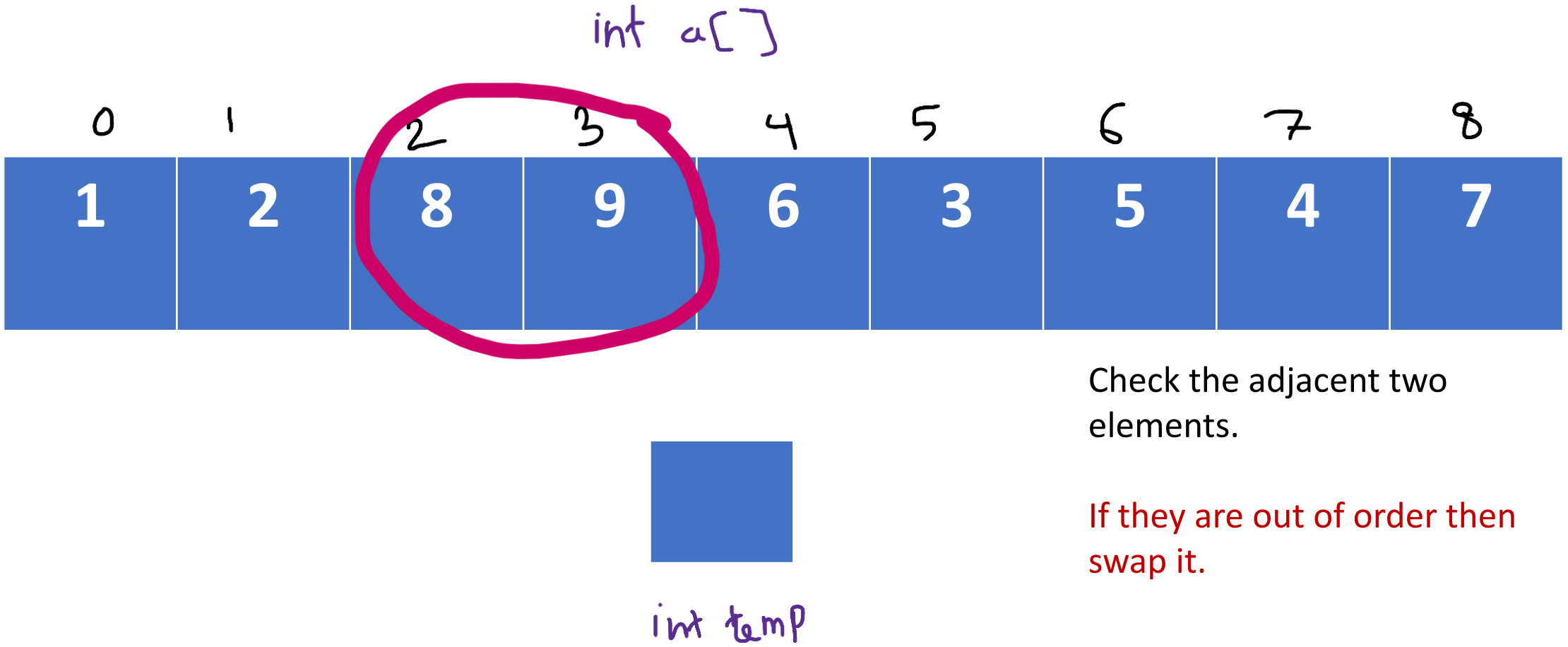


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

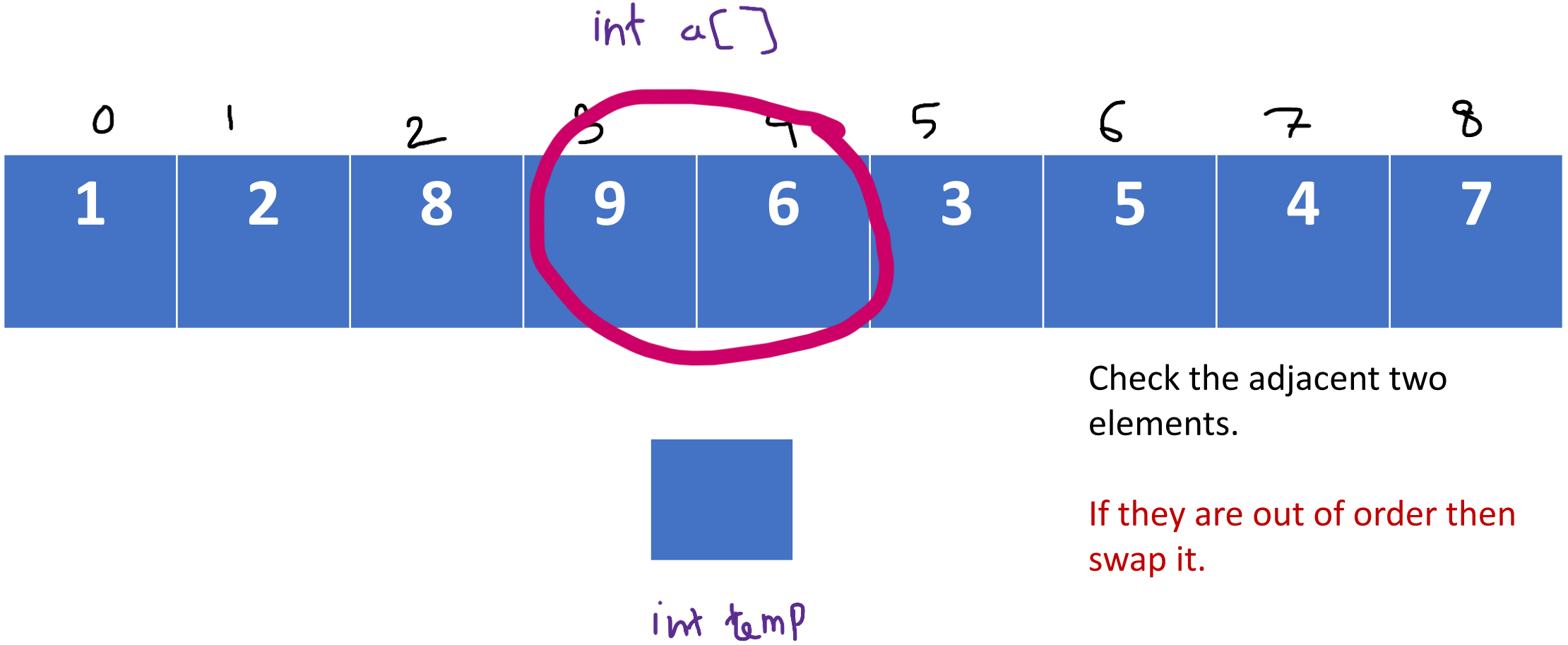


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

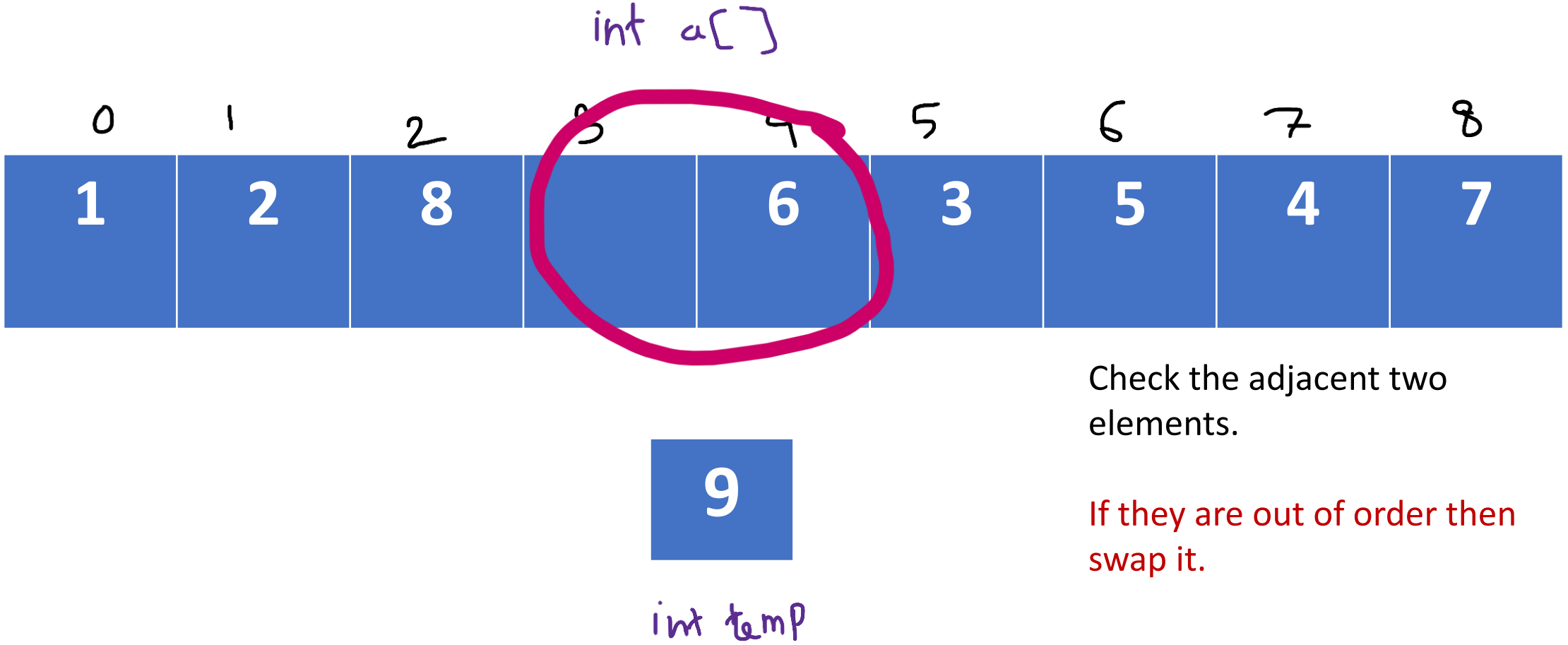


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

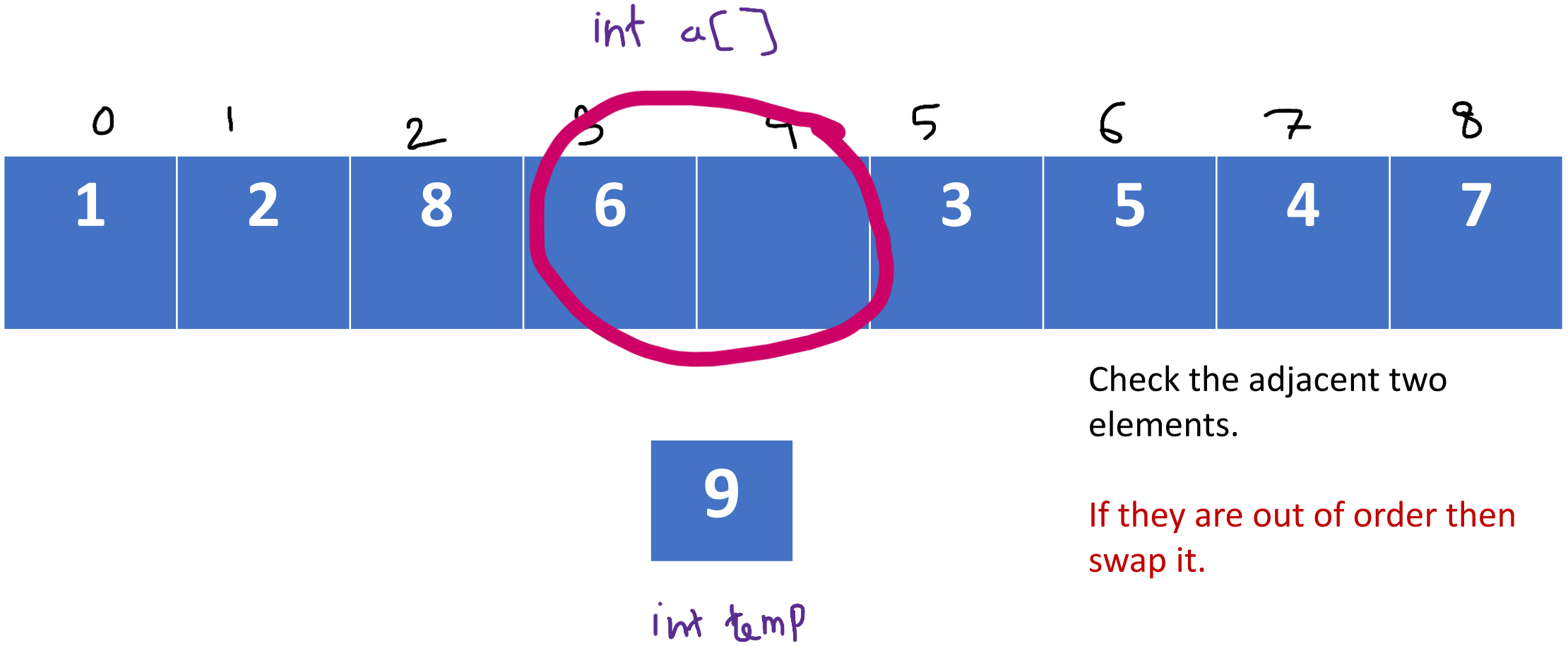


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

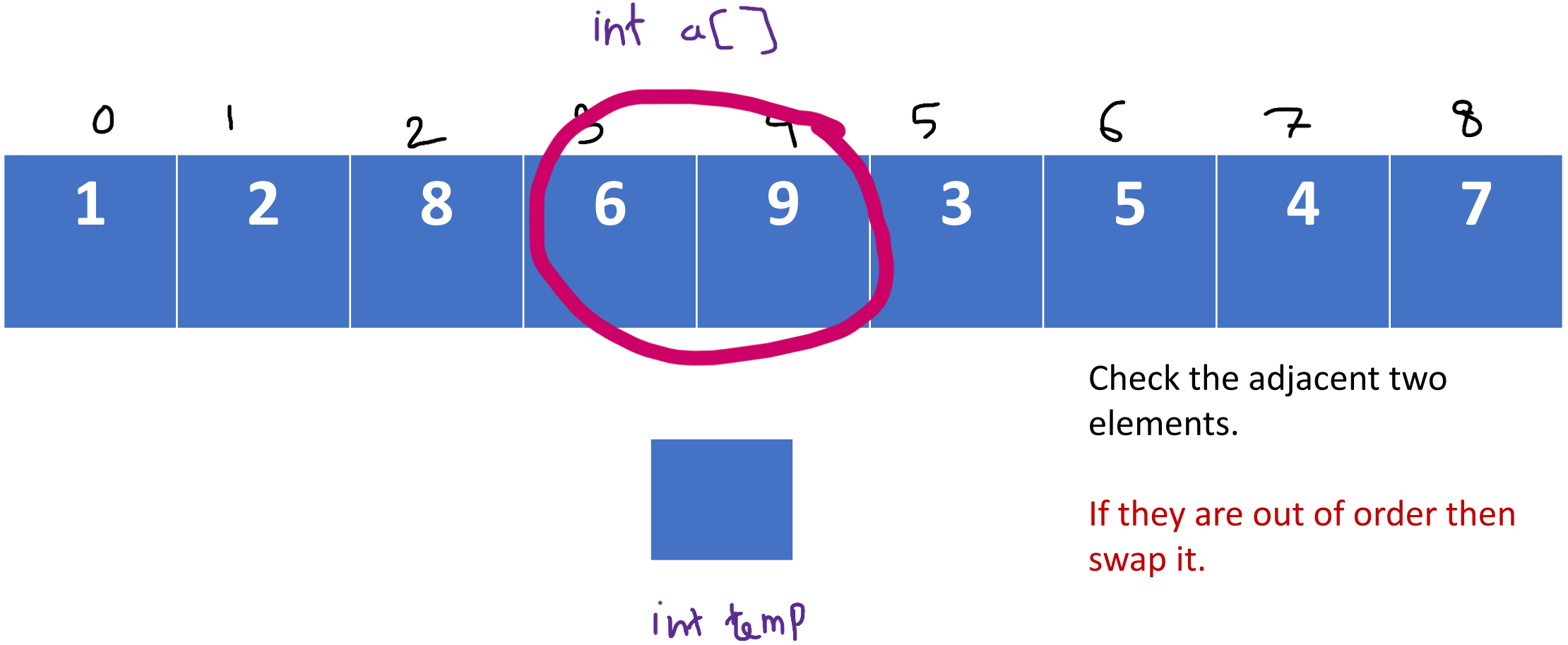


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

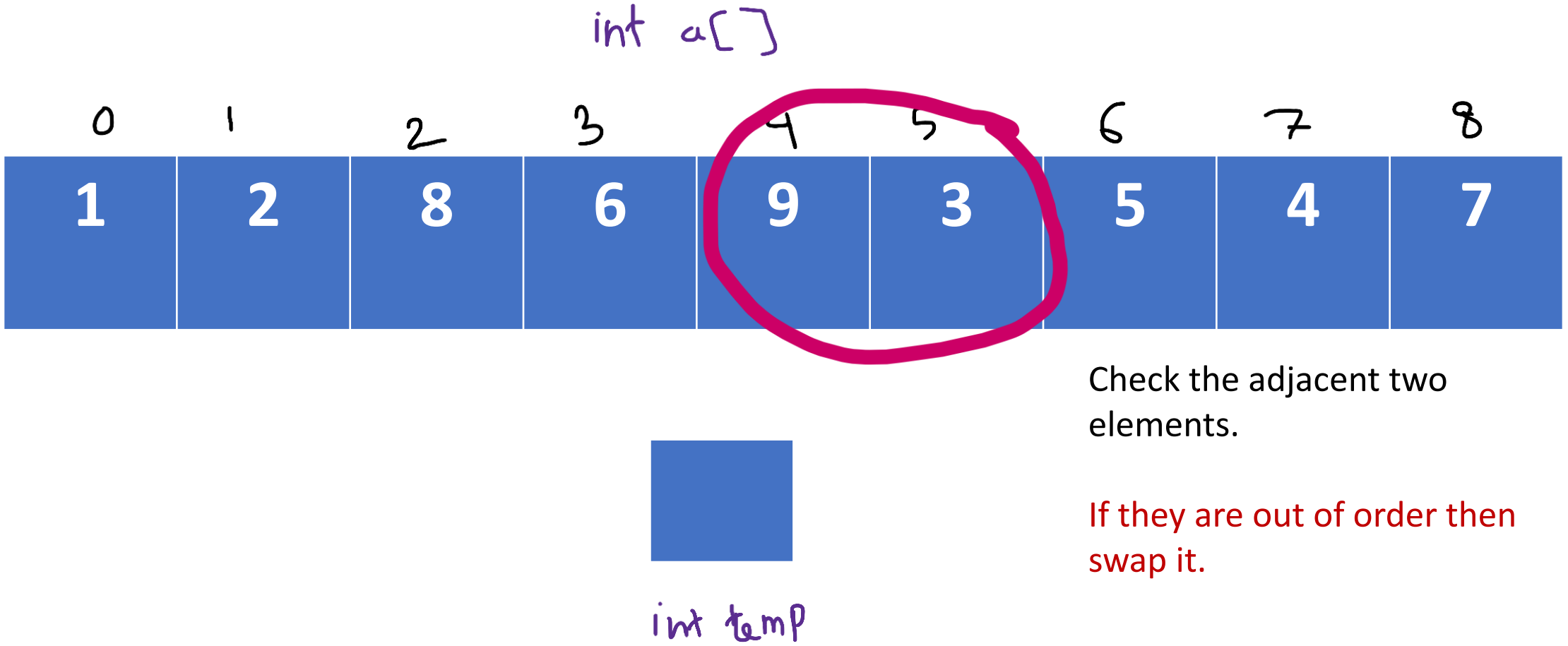


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

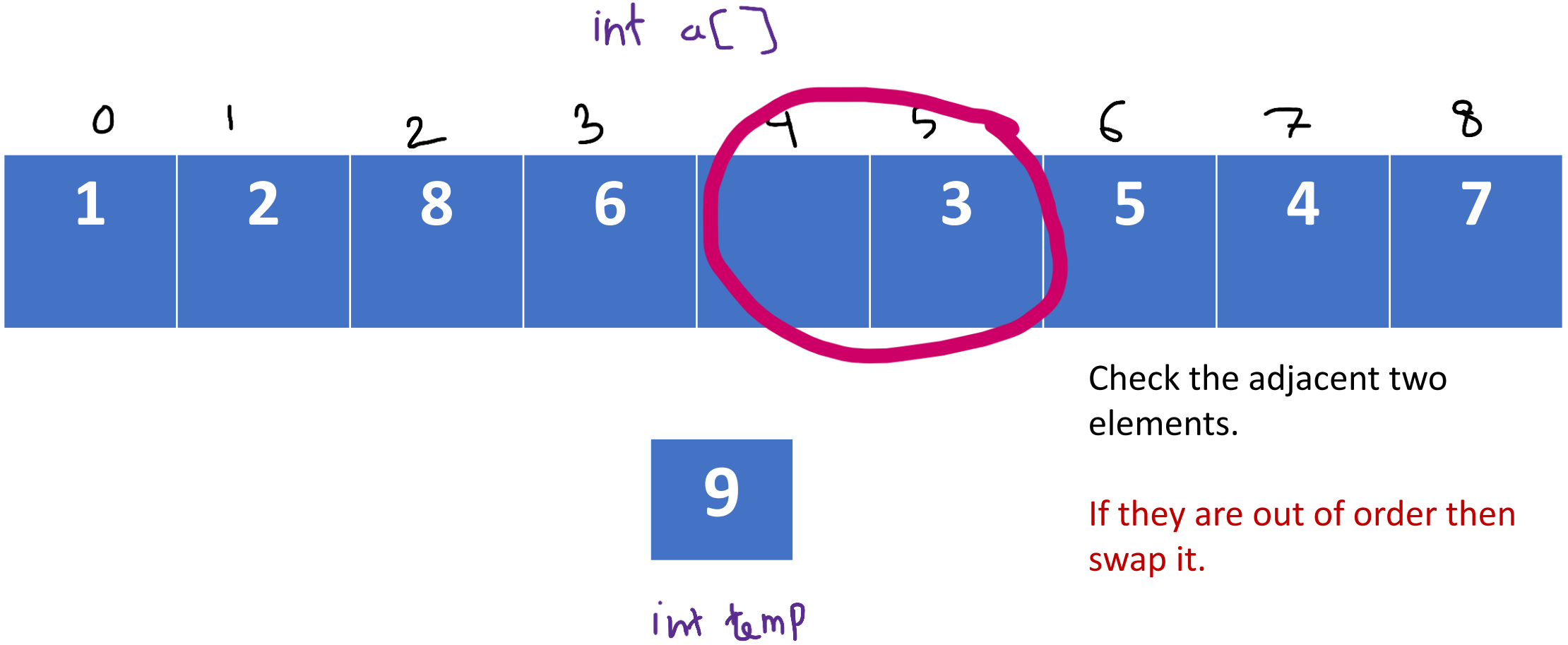


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

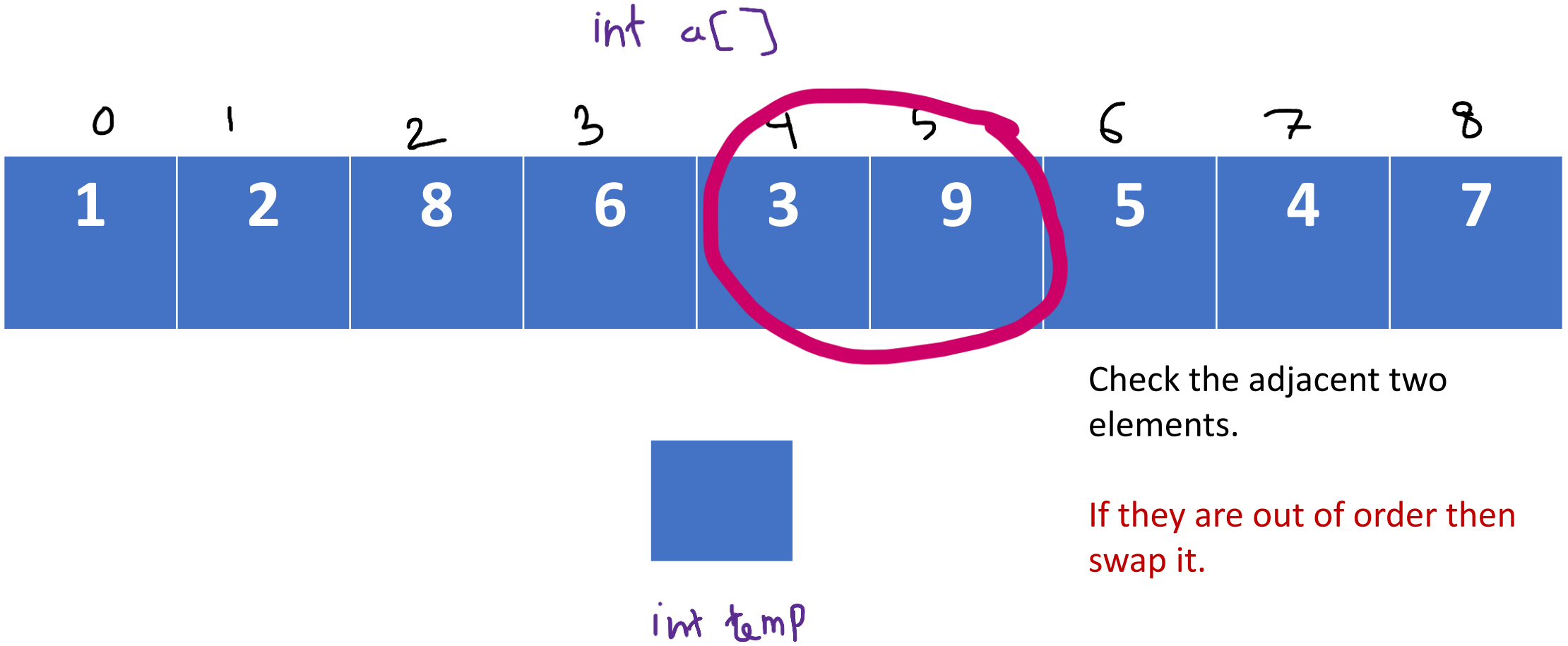


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort

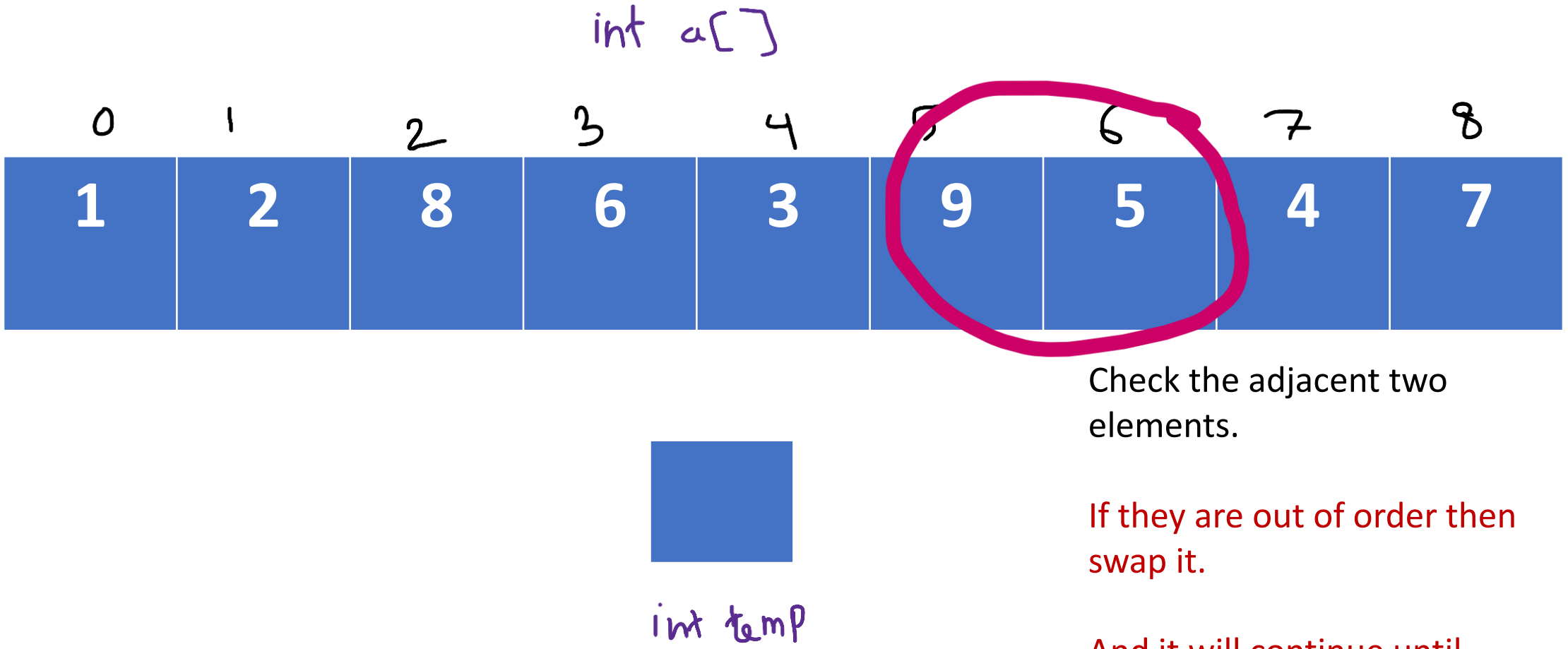


Check the adjacent two elements.

If they are out of order then swap it.

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble Sort



Check the adjacent two elements.

If they are out of order then swap it.

And it will continue until every item is sorted

https://www.youtube.com/watch?v=Dv4qLJcxus8

Bubble sort code

```
Public static void bubblesort(int array[]){

        for( int i=0; i<array.length-1; i++){
                for(int j=0; j< array.length-i-1; j++){
                        if (array[j]>array[j+1]{     // for ascending order
                                int temp = array[j];
                                array[j]=array[j+1];
                                array[j+1]=temp;
                        }
                }
        }
}
```

O(n^2)

What is the space complexity of selection,insertion and bubble sort?

- O(1) or constant space because all of them doesn’t require any additional space rather than a “temp” for swapping.
- Items can be sorted staying in their place.

Insertion sort is better for small or partially sorted lists, while selection sort is better for small datasets and when memory space is limited

Reading (The comparison)

- https://www.geeksforgeeks.org/comparison-among-bubble-sort-selection-sort-and-insertion-sort/