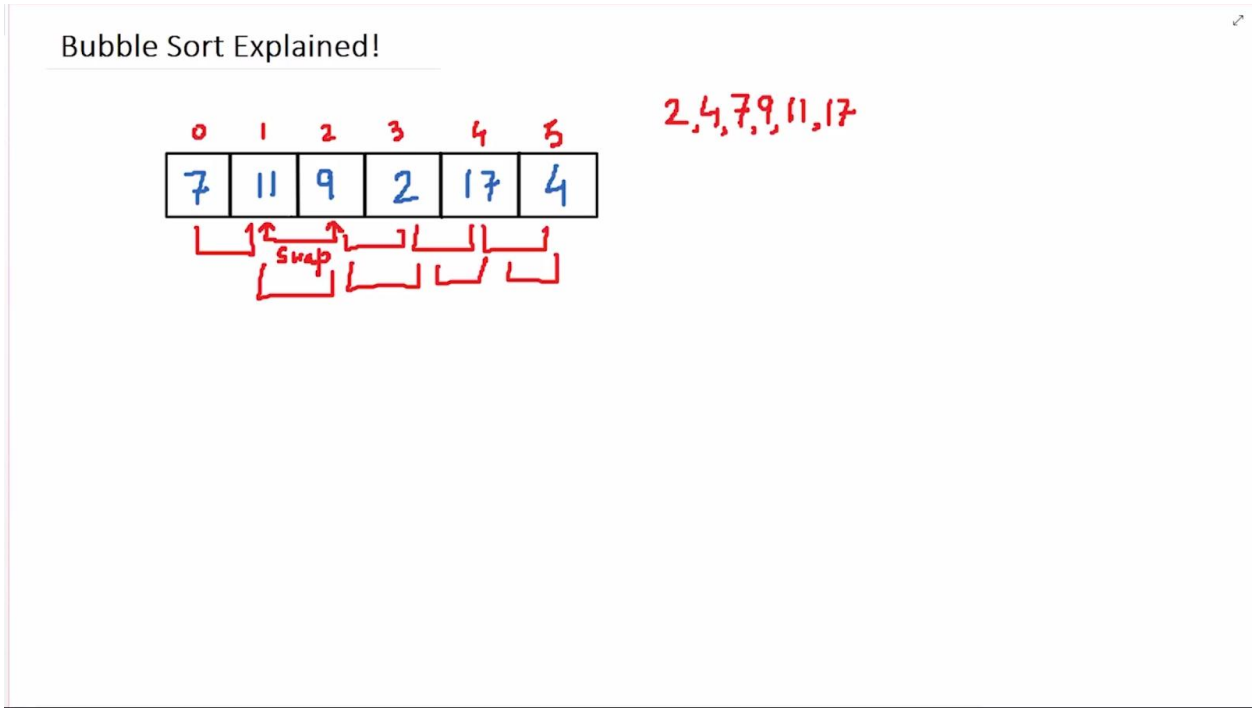
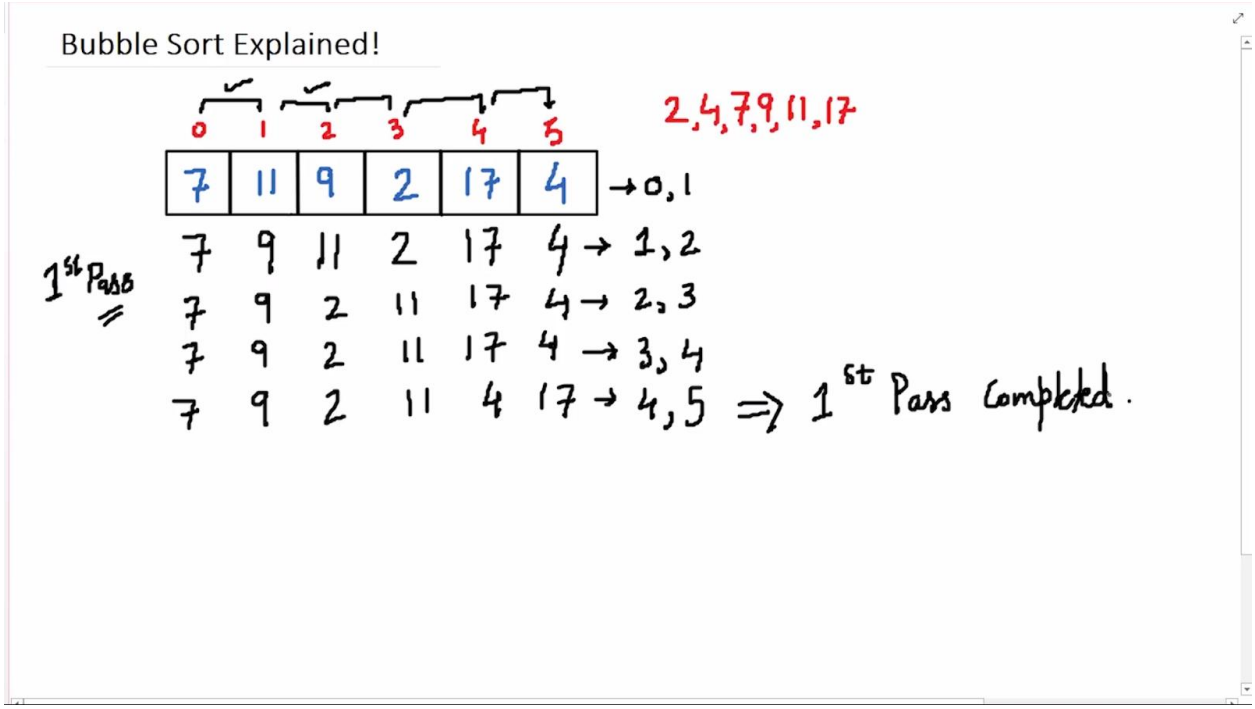
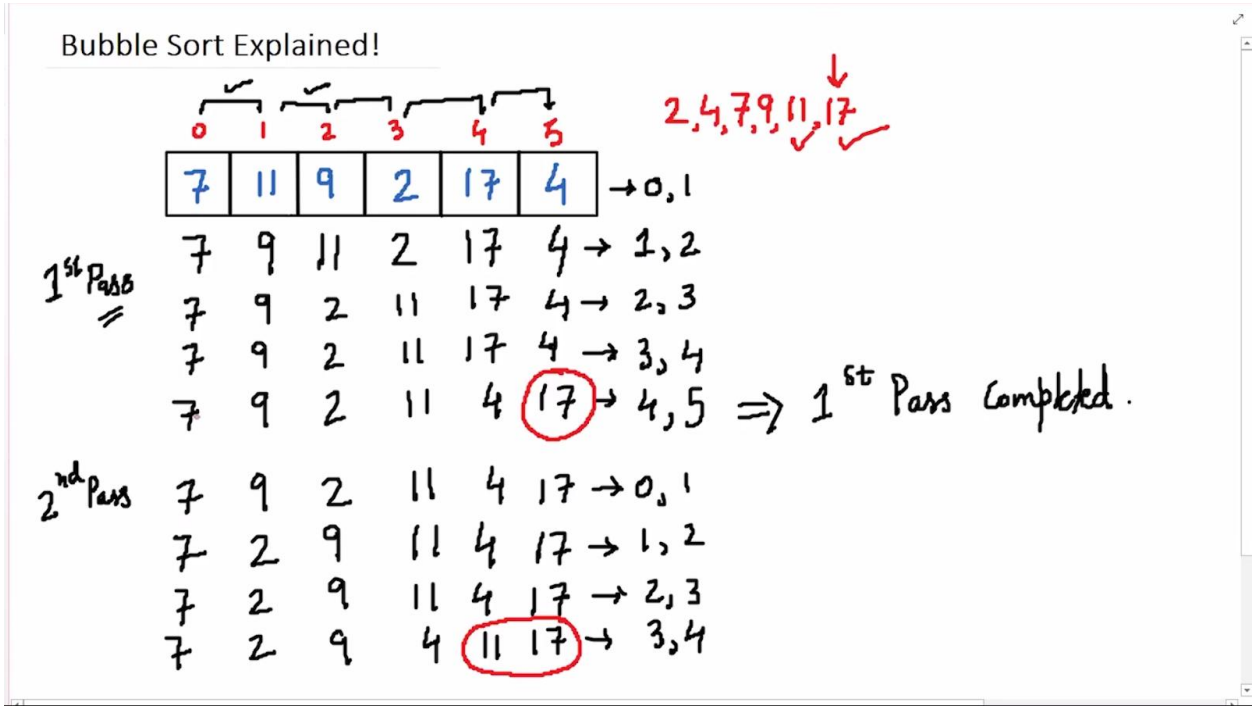


1st pass



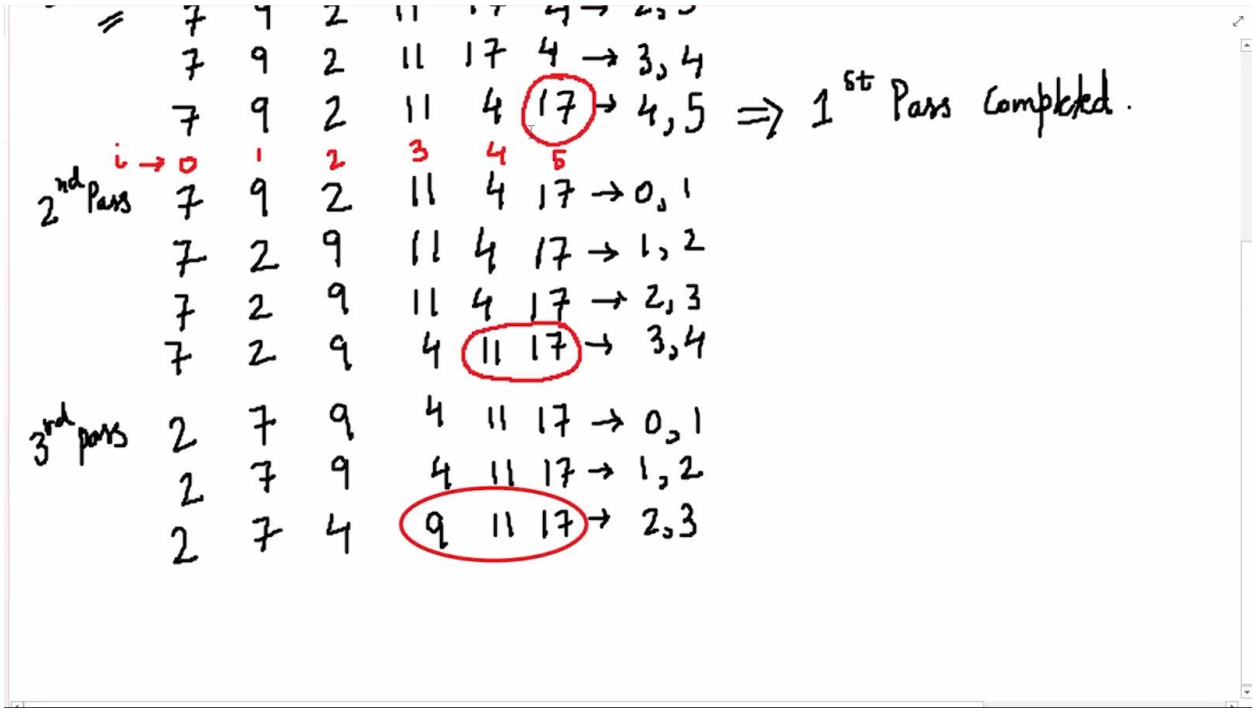
By 1st pass, → largest element will be placed at last position. Since last element is sorted so, in next pass we only have to sort (size – 1) elements

2nd pass

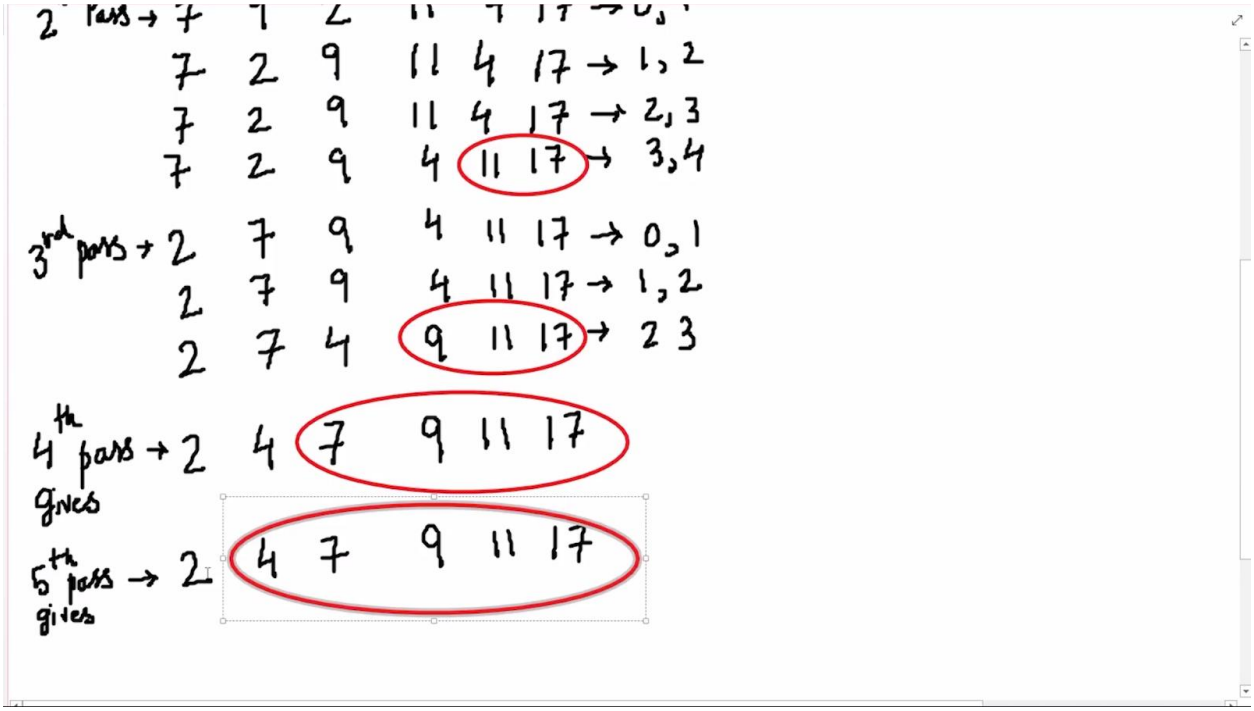


here last 2 elements are sorted so, in next pass we only have to do sorting for (size – 2) elements

3rd pass

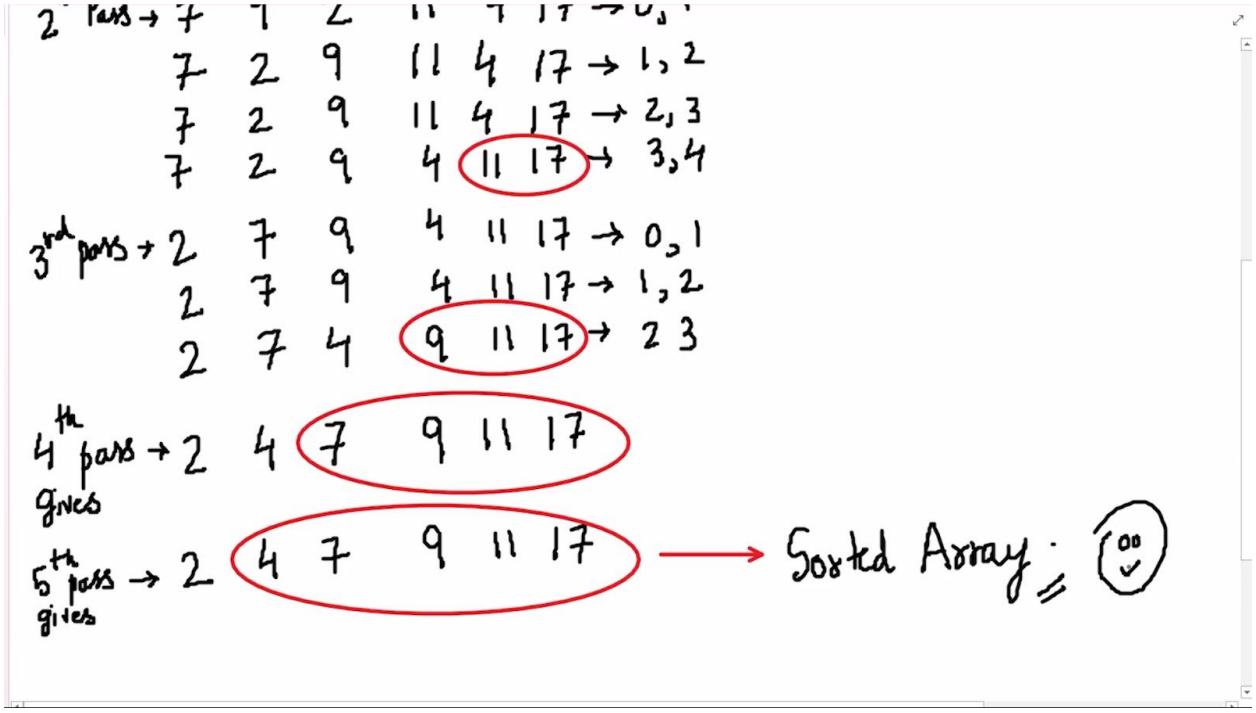


and similarly for others



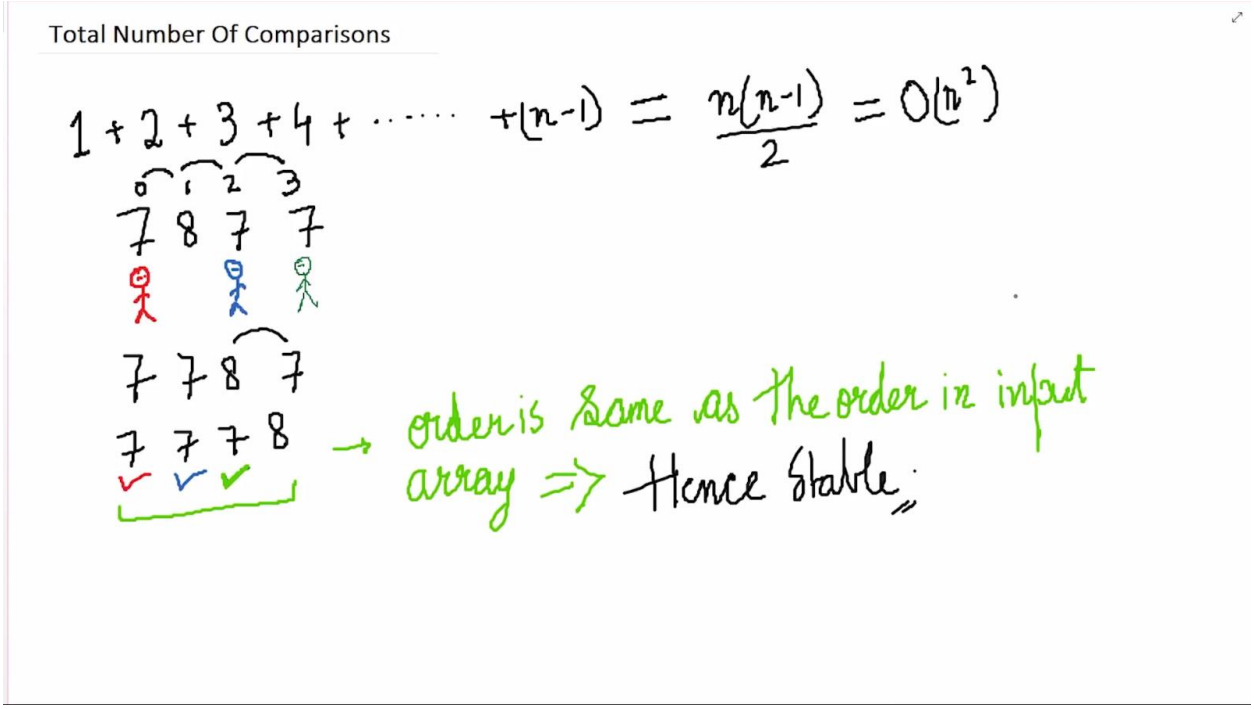
Notes: for size of array = n

1. No. of passes required = n – 1
2. Total no. of comparison / No. of possible swaps = (n - 1) + (n – 2) + (n – 3) + … + 2 + 1 = $\boldsymbol{n * \frac{n-1}{2}}$ = **$O(n^2)$ → time complexity**
3. No. of comparison / No. of possible swaps in 1st pass = n – 1
4. No. of comparison / No. of possible swaps in 2nd pass = n – 2
   … Similarly, for other passes …



Bubble sort is

1. **stable algorithm**
2. **not a recursive algorithm**
3. **by default it is not adaptive but it can be made adaptive (**in 1st pass if there is no successful swaps**)**



It is k/a Bubble sort because lighter element move to left side and heavier element move to right side similar to stone thrown in water

For already sorted array → time complexity = O(n)