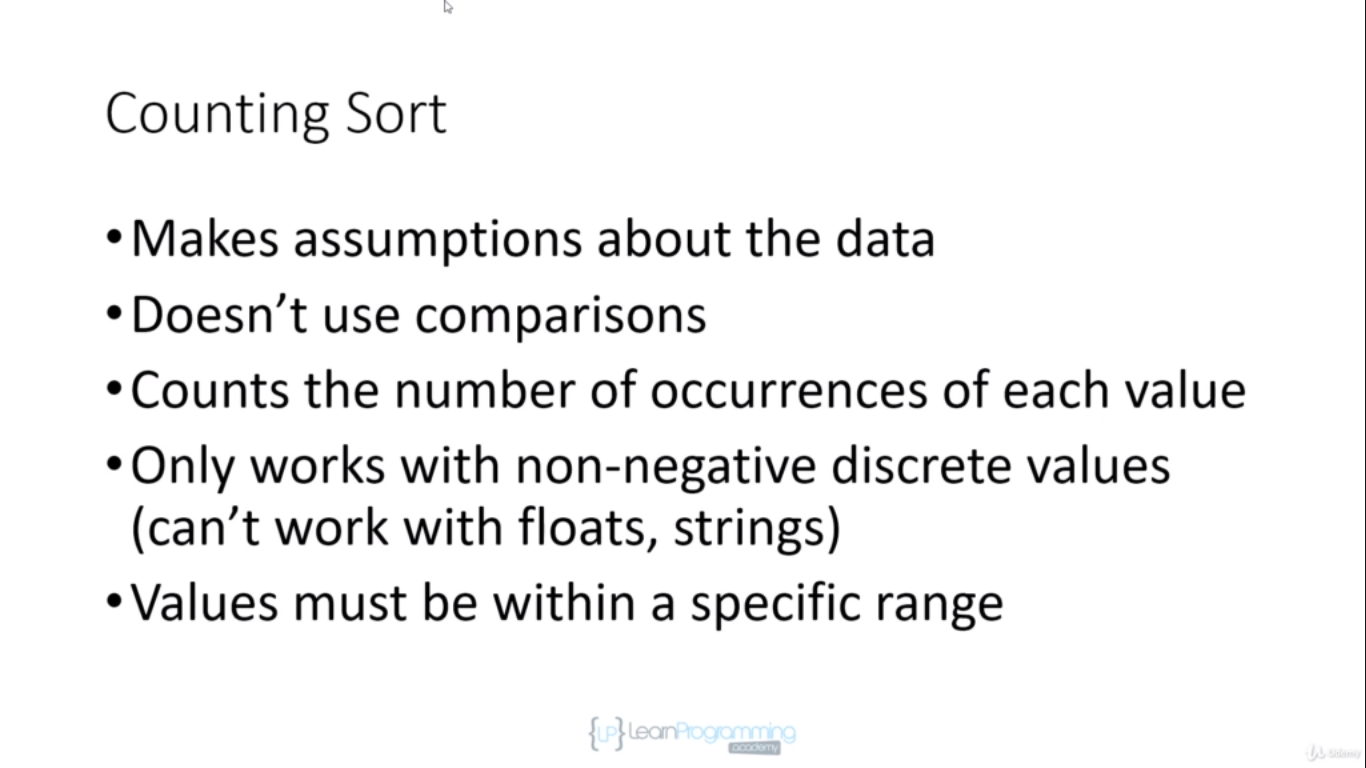
**Counting Sort**

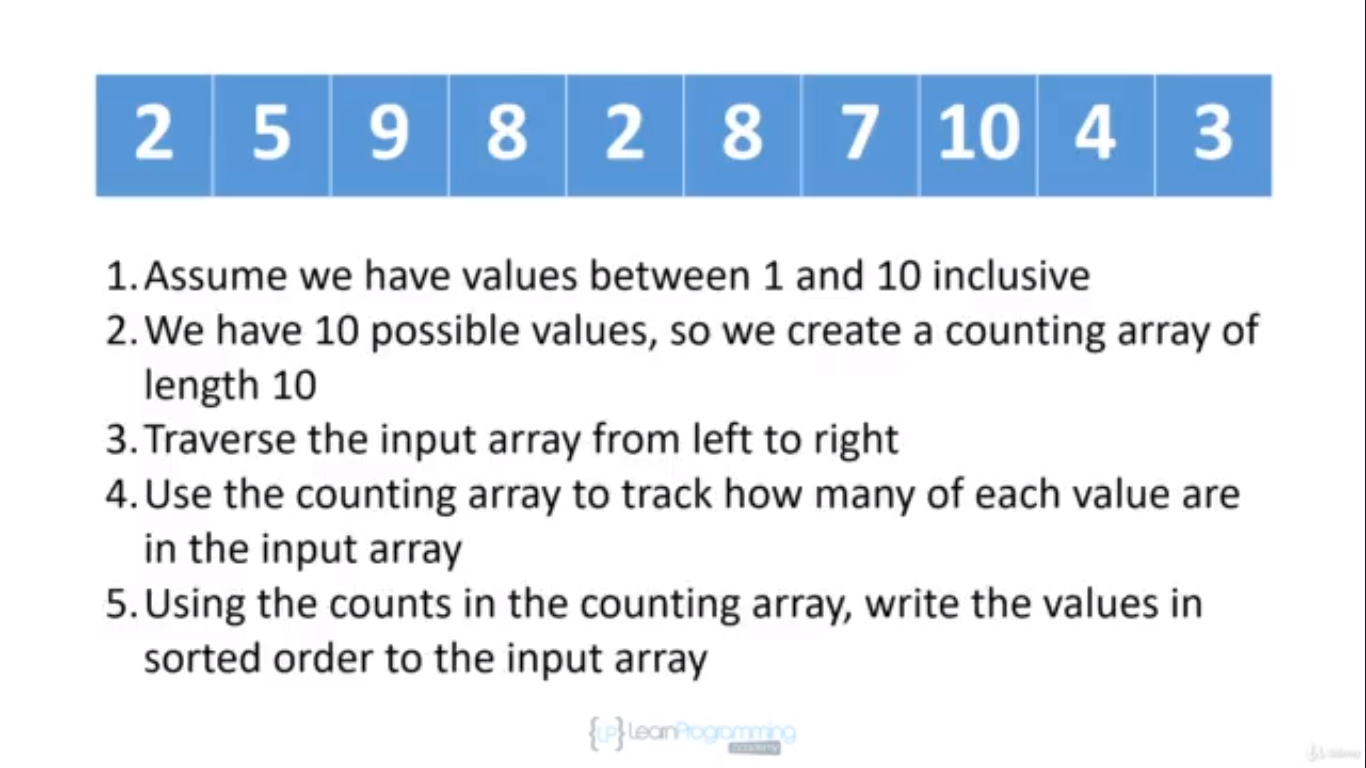
Algorithms we have seen till now don’t make assumptions about data. Now we have algorithm that are making assumptions about data they are sorting**.**

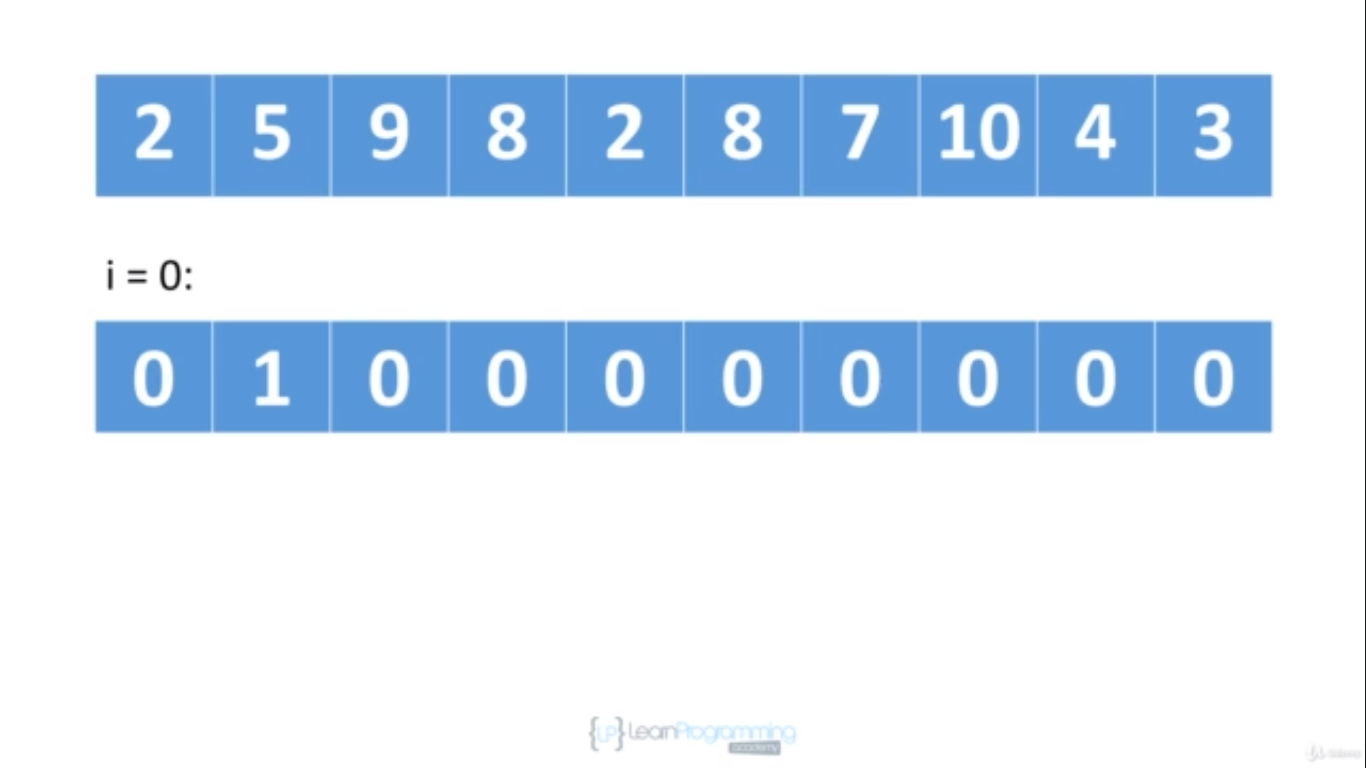


It will be used to sort whole numbers.

Values must be within a specific range, and range should be reasonable it can’t be huge, we can’t use this sort to sort elements between 1 and million.

We are going to use different array and assume





Position 0 will hold the number of 1 we find, Position 1 will hold the number of 2 we find, Position 2 will hold the number of 3 we find, Position 3 will hold the number of 4 we find and so on



For I =1 ^



For I =2^



For i=3^



For i=4^

We found another 2 so we increment value at counting array at position 1 to plus 1 which now becomes 2.



For i=5, we found 8 again so we increment value at position 7 in counting array to plus1 which is now 2.



For i=6^



For i=7 we found 10 so we increment position 9 to plus 1 in counting array as we can see above



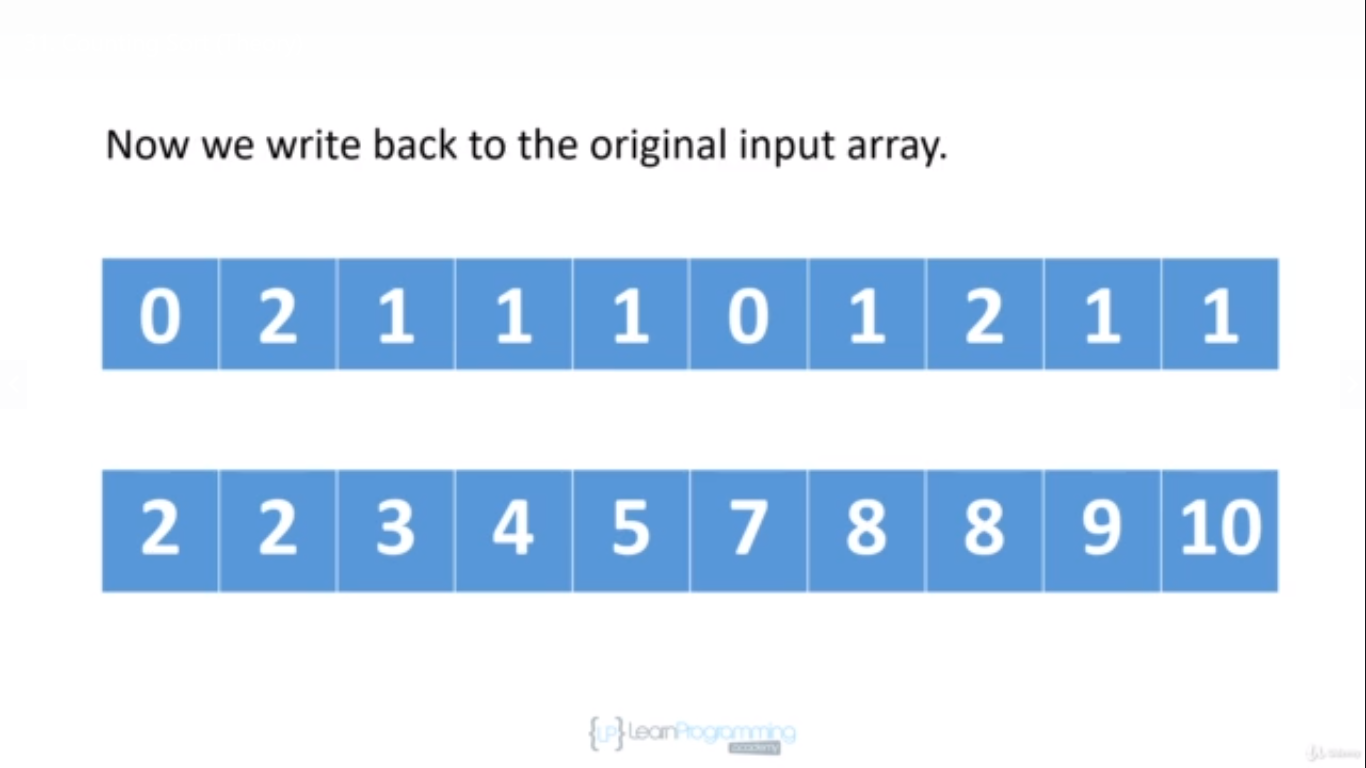
For i=8 we found 4 so we increment position 3 to plus 1 in counting array which is now 1 as seen above.



For i=9 we found 3 so finally we increment position 2 at counting array to plus1 so now value becomes 1 at position 2 as seen above

At this point we have successfully traversed our array and counted each element value occurrences we see.

Now we are going to write back original value to original input array.

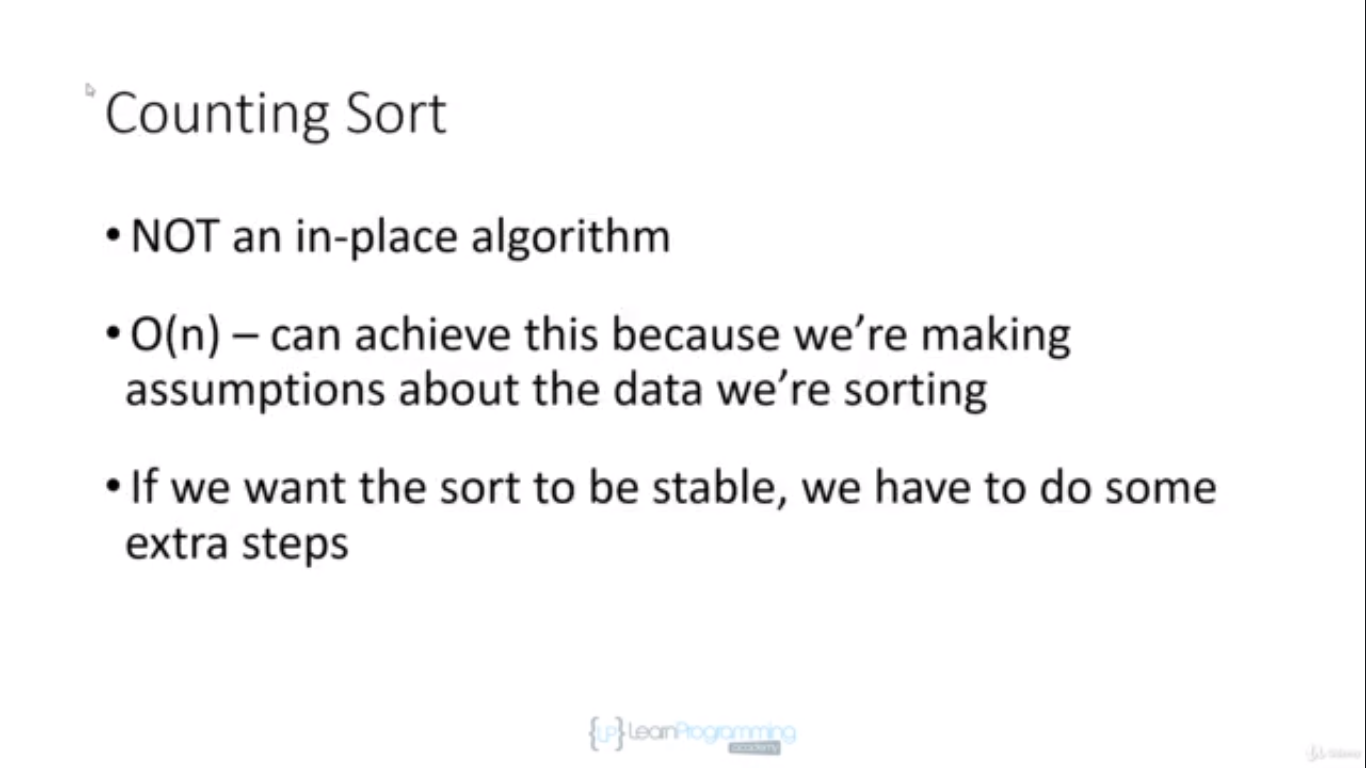


Counting array is at the top and input array is at the bottom

and so at this point we're gonna write two 2's into the input array and then there's one 3 so we'll write one 3 to the input array. There's one 4 so we'll write one 4 to the input array. There's one 5 so we'll write one 5 to the input array. There are zero 6, so we don't do anything. There is one 7, so we write one 7 to the input array. There are two 8, so we're gonna write two 8's to the input array. There's one 9, so we write one 9 to the input array and finally there's one 10 and so we write one 10 to the input array.

So essentially counting sort has two phases. The first phase is we traverse the input array and we count how many of each value we have and then in the second phase using the counting array, we write values back into the input array

and so as you can see our input array is now sorted and we didn't do any comparisons. We didn't compare any of the elements against each other. We're just merely counting, how many of each value we have and that's why the algorithm is called counting sort.



Not an in-place algorithm, best suited for elements whose range of values is equivalent to no of values which you want to sort.

Not suitable for elements whose range is between 1 to 10000 but array is of ten elements out of that range so that case counting array we have to create is of 10000 elements to sort 10 elements which is not recommended.

To make it stable we can use linkedlist but that will be covered later.

