Counting Sort

Key Points

The O(N) sort.

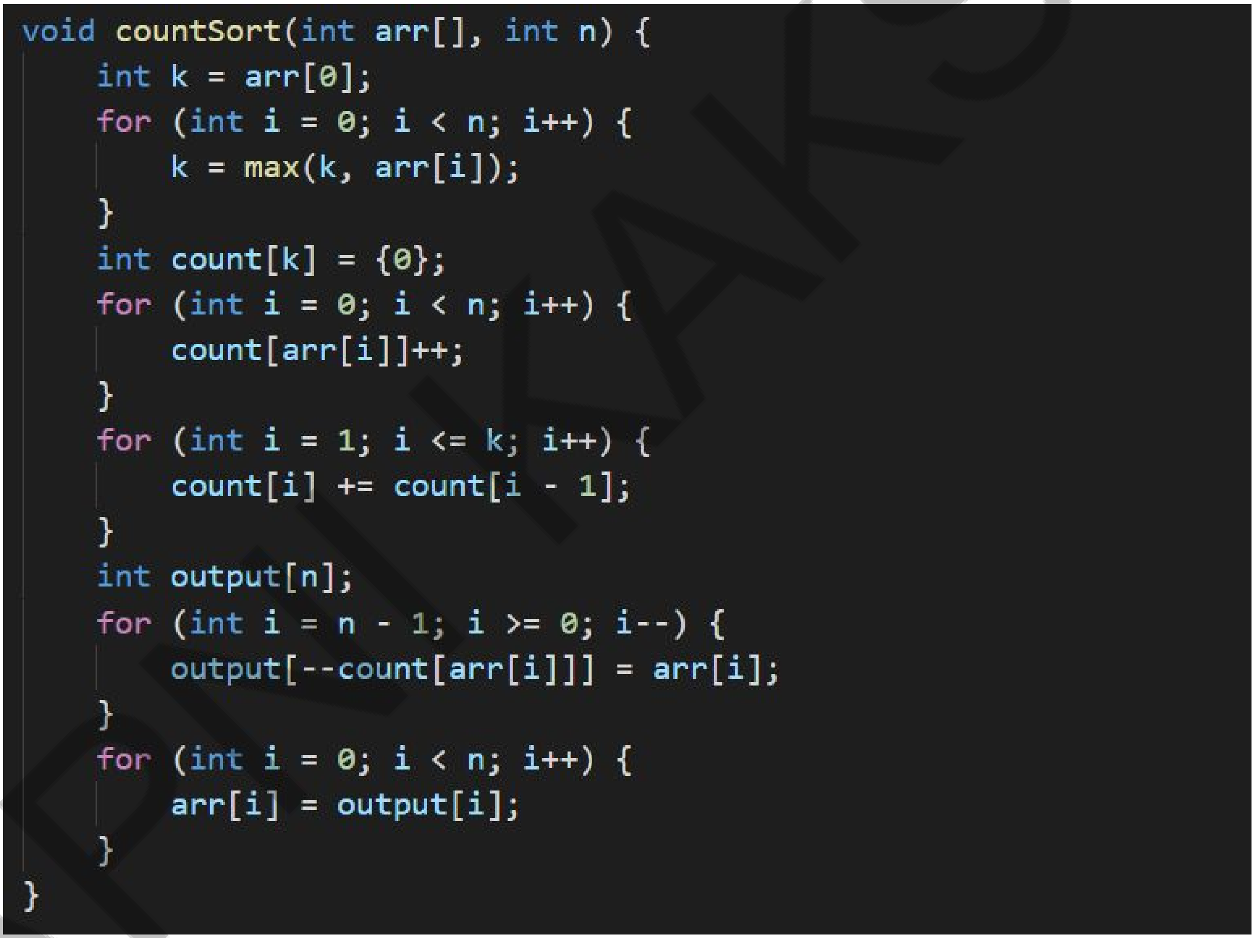
Note: Only limited to small positive integers only.

Input: Given an array.

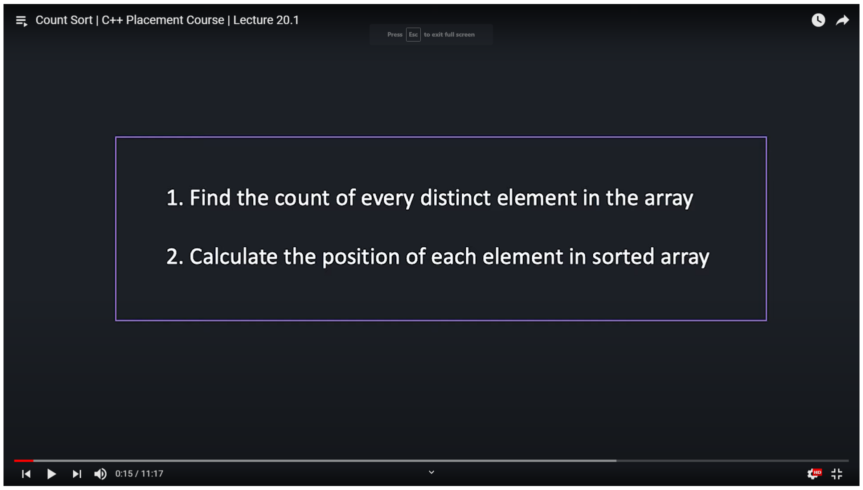
Output: print the sorted array.

Time Complexity: O(N)

Space Complexity: O(max(Arr))

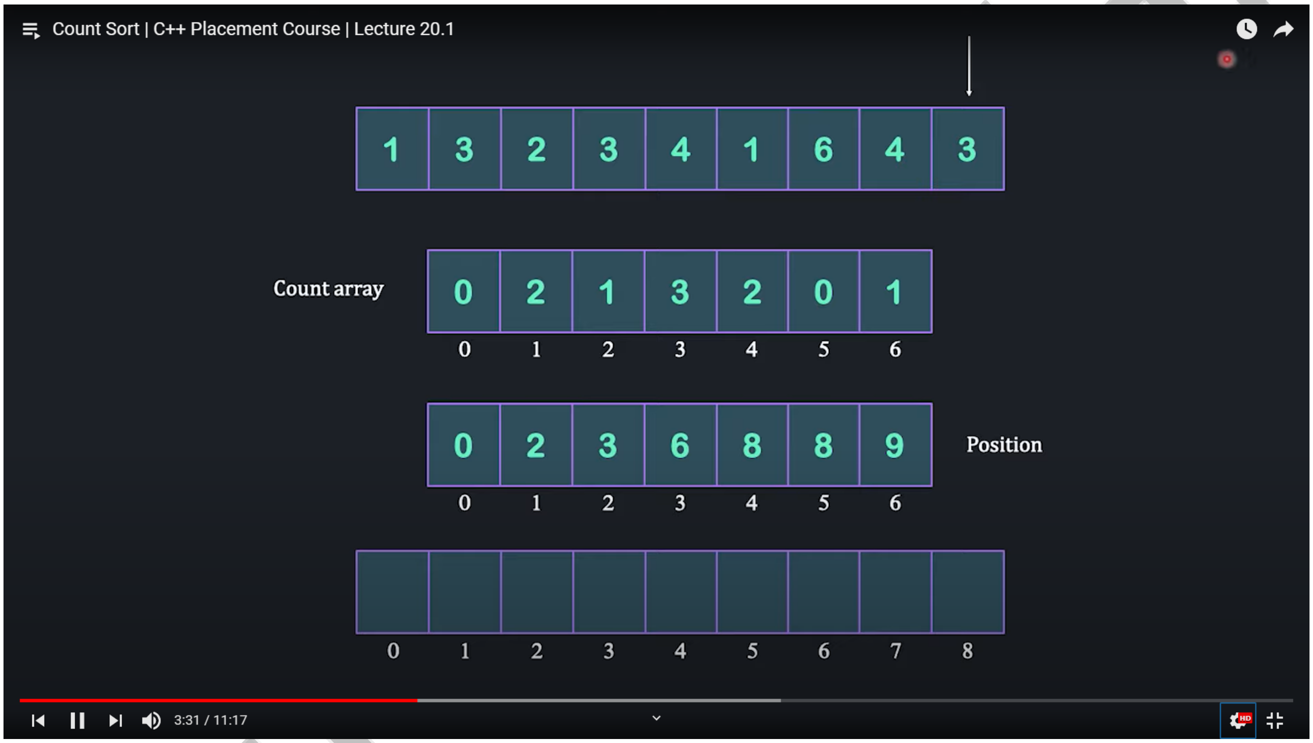


```
void countSort(int arr[], int n) {
    int k = arr[0];
    for (int i = 0; i < n; i++) {
        k = max(k, arr[i]);
    }
    int count[k] = {0};
    for (int i = 0; i < n; i++) {
        count[arr[i]]++;
    }
    for (int i = 1; i <= k; i++) {
        count[i] += count[i - 1];
    }
    int output[n];
    for (int i = n - 1; i >= 0; i--) {
        output[--count[arr[i]]] = arr[i];
    }
    for (int i = 0; i < n; i++) {
        arr[i] = output[i];
    }
}
```


Count Sort | C++ Placement Course | Lecture 20.1
Press Esc to exit full screen
0:15 / 11:17

1. Find the count of every distinct element in the array

2. Calculate the position of each element in sorted array



1	3	2	3	4	1	6	4	3

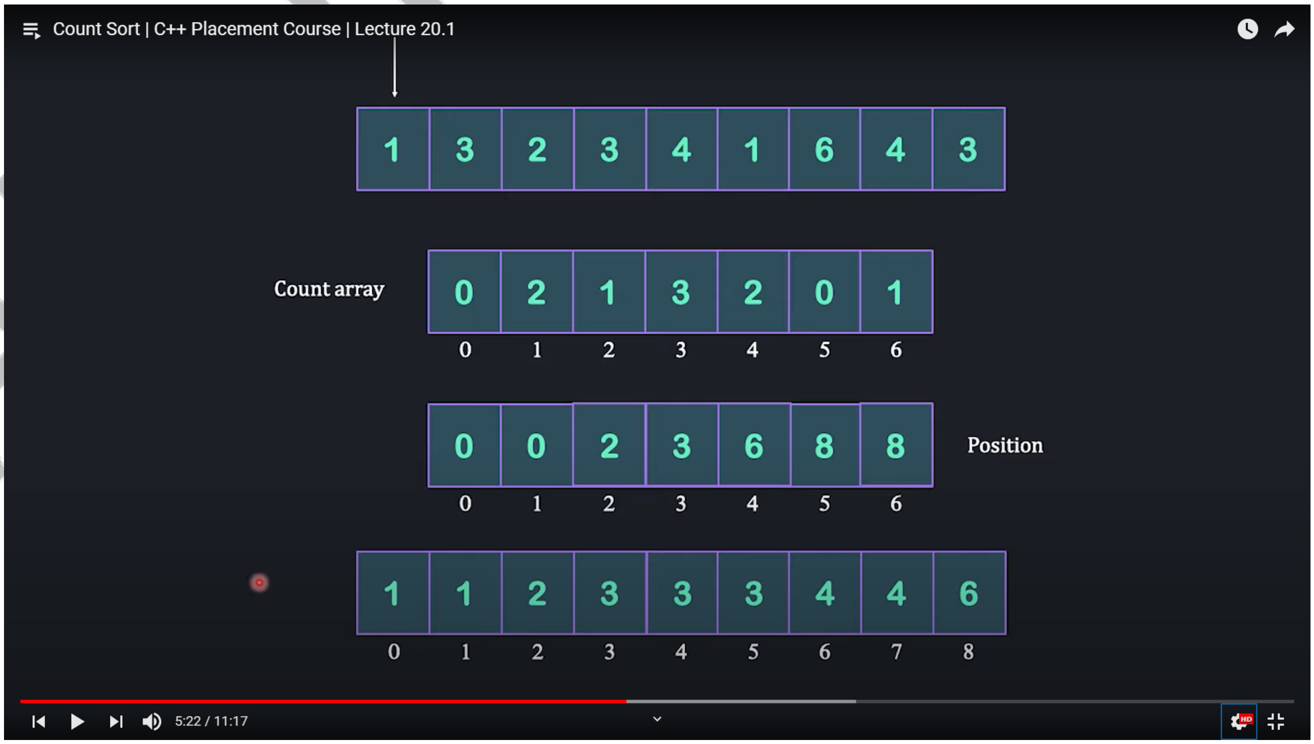
Count array

0	2	1	3	2	0	1
0	1	2	3	4	5	6

Position

0	2	3	6	8	8	9
0	1	2	3	4	5	6

0	1	2	3	4	5	6	7	8



1	3	2	3	4	1	6	4	3

Count array

0	2	1	3	2	0	1
0	1	2	3	4	5	6

Position

0	0	2	3	6	8	8
0	1	2	3	4	5	6

1	1	2	3	3	3	4	4	6
0	1	2	3	4	5	6	7	8