Count Sort Algorithm

The count sort algorithm assumes that the input is relatively smaller so the algorithm is as follows –

Step 1 – Maintain two arrays, one with the size of input elements without repetition to store the count values and other with the size of the input array to store the output.

Step 2 – Initialize the count array with all zeroes and keep the output array empty.

Step 3 – Every time an element occurs in the input list, increment the corresponding counter value by 1, until it reaches the end of the input list.

Step 4 – Now, in the output array, every time a counter is greater than 0, add the element at its respective index, i.e. if the counter of '0' is 2, '0' added at the 2nd position (i.e. 1st index) of the output array. Then decrement the counter value by 1.

Step 5 – Repeat Step 4 until all the counter values become 0. The list obtained is the output list.

```
COUNTING-SORT(A, n, k)
 1let B[1 : n] and C [0 : k] be new
   arrays
 2for i = 0 to k
 3     C [i] = 0
 4for j = 1 to n
 5     C [A[j]] = C [A[j]] + 1
 6// C [i] now contains the number of elements equal to i.
 7for i = 1 to k
 8     C [i] = C [i] + C [i – 1]
 9// C [i] now contains the number of elements less than or
   equal to i.
10// Copy A to B, starting from the end of A.
11for j = n downto 1
12     B[C [A[j]]] = A[j]
13     C [A[j]] = C [A[j]] – 1          // to handle duplicate
                                           values
14return B
```

Analysis

The time complexity for the counting sort algorithm is **O(n + k)** time.

And **O(n)** when k ≤ n.