Sorting by Counting
or
Counting Sort

Algorithm Problem Solving (APS)
17ECSE309

By,
Anup V T
01FE15BEC032

Counting Sort:

- Counting sort assumes that each of the n input elements is an integer in the range 0 to k. that is n is the number of elements and k is the highest value element.

- Consider the input set : 4, 1, 3, 4, 3. Then n=5 and k=4 .

- Counting sort determines for each input element x, the number of elements less than x. And it uses this information to place element x directly into its position in the output array. For example if there exits 17 elements less that x then x is placed into the 18th position into the output array.

- The algorithm uses three array:
  Input Array : A[1..*n*] to store input data where A[j] ∈ {1, 2, 3, …, *k*}
  Output Array : B[1..*n*] to finally store the sorted data
  Temporary Array : C[1..*k*] to store data temporarily

Algorithm of Counting Sort:

Counting-Sort (A, B, k) //Function

1. Let C[0…..k] be a new array
2. for i=0 to k

        C[i]= 0;
3. for j=1 to A.length or n

        C[A[j]] = C[A[j]] + 1;
4. for i=1 to k

        C[i] = C[i] + C[i-1];
5. for j=n or A.length down to 1

        B[C[A[j]]] = A[j];

        C[A[j]] = C[A[j]] - 1;

Time Complexity : O(n+k)

Advantages:

- Simple to Code.
- High efficiency.
- Stable Sort.

  -A sorting algorithm is ***stable*** when numbers with the same values appear in the output array in the same order as they do in the input array.

Disadvantages:

- Requires (n+k) extra storage.
- Works well only for small numbers.

Applications:

- Often used as a subroutine in another sorting algorithm, radix sort , that can handle larger keys more efficiently.

References :

https://www.cs.usfca.edu/~galles/visualization/CountingSort.html

https://medium.com/basecs/counting-linearly-with-counting-sort-cd8516ae09b3

https://www.hackerearth.com/practice/algorithms/sorting/counting-sort/tutorial/

https://brilliant.org/wiki/counting-sort/

http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/Sorting/countingSort.htm