DAA Lab - Session 10 - Distribution Counting Sorting

**Space and Time Tradeoff:** Implementation of Distribution Counting Sorting

**Problem Definition:** Sort an array of integers using Distribution Counting Sorting algorithm (and also using Merge Sort algorithm) in nondecreasing order. Print the sorted array and the duration of execution of both methods.

**Input:** Input begins with n (1 ≤ n ≤ 220) of number of records indicating the size of the input array. The following n lines has an integer per line. Integer values are 32-bit signed integers.

**Output:** Print the sorted array one value per line. Print the duration of execution of Distribution Counting Sorting method followed by the Mergesort method (keep precision upto 6 decimal places).

**Sample Input:**

5

20

40

20

10

30

**Sample Output:**

10

20

20

30

40

30

0.000000 sec.

0.000000 sec.

**Sample Input:**

3

400100

201030

800909

**Sample Output:**

201030

400100

800909

0.000000 sec.

0.000000 sec.

**Algorithm MergeSort(A[0..n-1])**

//Sorts array A[0..n-1] by recursive Merge Sort

//Procedure Merge(A[0..n-1], m) merges two

// sorted subarrays A[0..m-1] and A[m..n-1]

// into a sorted array A[0..n-1].

**if(n ≤ 1)return**

**m = ⌊n/2⌋**

**MergeSort(A[0..m-1])**

**MergeSort(A[m..n-1])**

**Merge(A[0..n-1], m)**

**Algorithm Merge(A[0..n-1], m)**

//Merges two sorted arrays A[0..m-1] and A[m..n-1] into

//the sorted array A[0..n-1]

**i ← 0, j ← m, k ← 0**

**while(i < m and j < n) do**

**if(A[i] ≤ A[j]) B[k] ← A[i]; i ← i+1**

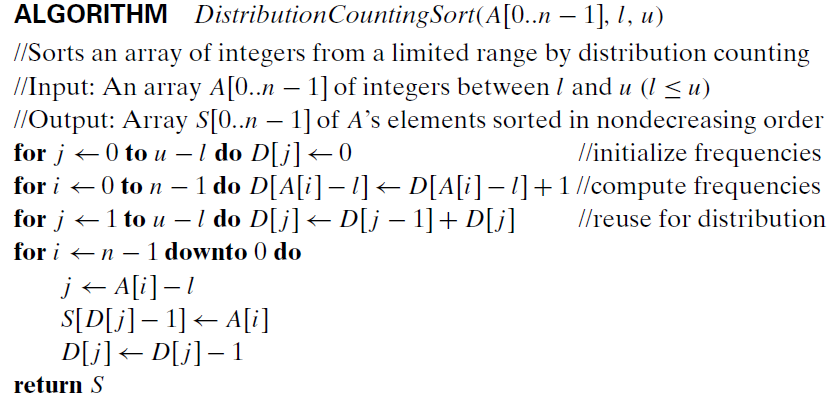
**else B[k] ← A[j]; j ← j+1**

**k ← k+1**

**if(j = n) Copy A[i..m-1] to B[k..n-1]**

**else Copy A[j..n-1] to B[k..n-1]**

**Copy B[0..n-1] to A[0..n-1]**

****

**Practice Problems:**

1. Based on the range of input values and the size of the input array, sort with the best of two algorithms.