# California State University, Fresno

# DEPARTMENT OF COMPUTER SCIENCE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class: | **Algorithms & Data Structures** | | | Semester: | **Fall 2021** |
|  | | | | | |
| Points |  | Document author: | **Saishnu Ramesh Kumar** | | |
|  | Author’s email: | **saishnu\_rk@mail.fresnostate.edu** | | |
| Laboratory number: | **Section 1, 11am to 12:50pm** | | |
|  | | | | | |

**1. Statement of Objectives**

For this assignment, we were tasked to create two codes, one for counting sort and the other for radix sort. Both of these sorting algorithms have their own time complexity, where counting sort’s is O(n + k), if k = O(n). The time complexity for radix sort is O(d \* (n + k)) where d is equal to the number of digits of the maximum input number and k is the total range of numbers from 0 to 9. We were also asked to record the execution times for both programs to see which is faster and more efficient as well.

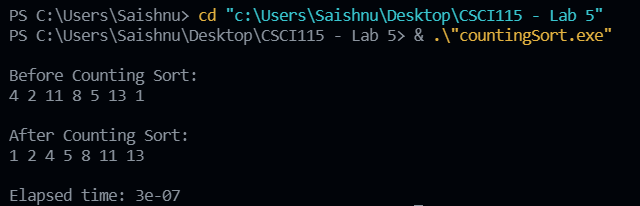
**2. Experimental Procedure**

For the counting sort program, I created 3 functions: printArray, countingSort, and main. The countingSort function contained the code where the array would get sorted. This algorithm required a number of for loops because it was running through the list multiple times, first to locate the largest element in the list to even putting the arrays in an auxiliary and getting the sorted list later on. The count in this represent the auxiliary array where every number in this list was set to zero. Then a cumulative count was done to add the values of the array together. Then it starts to locate each index for the element from the original array in the main function and would then put it into the output array list and then assign the output array back to the original array so that it gets updated with the sorted list. In the main function, it will call the sorting function and the print array function for the output.

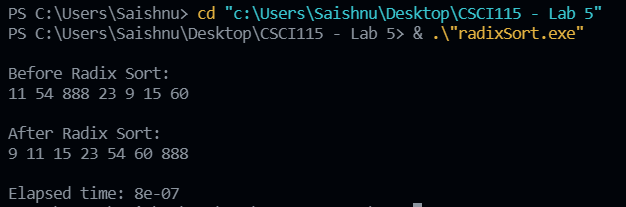
For the radix sort program, I created 5 functions: getMax, printArray, countingSort, radixSort and main. The radix sort algorithm somewhat takes certain parts that can be found in the counting sort algorithm as well. This algorithm will search for the number with the biggest digits in the list. As seen in the lab slides, it mentions that radix sort uses the counting sort as a subroutine to help perform the sorting and therefore a countingSort function was included because the concept of radix sort is similar to counting sort. There is a place variable in the function as well and that is for the grouping of the digits that have the same place value as the other elements. In the radixSort function, it gets the maximum element and is also calling the countingSort function to sort the elements according to their place value. The main function will call the sorting function as well as the print array function to get the final output.

**3. Analysis**

Counting Sort Screenshot Output:



Radix Sort Screenshot Output:



It can be seen that counting sort’s execution time is slightly faster than radix sort.

**4. Encountered Problems**

For this assignment, I encountered a few compiler errors as well as when the terminal did not output anything for the counting sort but that was due to a logical error, and it was fixed. I also tried to input a few large values for the counting sort algorithm, but it took a very long time to execute which is why I decided to change the values to smaller ones.

**5. Conclusions**

After doing this assignment, I think I have a fairly decent understanding of how counting and radix sorting algorithms work accordingly. Moreover, I learnt that counting sort is not very efficient when it comes to dealing with large values as compared to radix sort. Nonetheless, the theoretical aspect of it could be a little confusing at times but I would just need to revise and practice going through the notes as well as the codes once again.

**6. References**

1. Slides provided by TA during Lab Session
2. <https://www.programiz.com/dsa/counting-sort>
3. https://www.geeksforgeeks.org/radix-sort/
4. <https://www.programiz.com/dsa/radix-sort>