NAME: ADEBAYO ESTHER ADERONKE
REG NO: 24000144
COURSE TITLE: ADVANCED ALGORITHM
COURSE CODE: CSC 811

QUESTION: DESCRIBE TWO SORTING ALGORITHMS, STATING WHAT ALGORITHMIC PARADIGM THEY FOLLOW. STATE THEIR ADVANTAGES, DISADVANTAGES AND THEIR USE CASES.

1. COUNTING SORT: Counting Sort is a non-comparison-based sorting algorithm. It is particularly efficient when the range of input values is small compared to the number of elements to be sorted. The basic idea behind Counting Sort is to count the frequency of each distinct element in the input array and use that information to place the elements in their correctly sorted positions. Counting Sort follows the **Brute Force and Bucket-based algorithmic paradigms** because it relies on direct counting and indexing of elements.

For example, for input [1, 4, 3, 2, 2, 1], the output should be [1, 1, 2, 2, 3, 4]. The important thing to notice is that the range of input elements is small and comparable to the size of the array.
If the max element is of order more than n Log n where n is size of the array, then we can better sort the array using a standard comparison based sorting algorithm.

Counting Sort Algorithm:

- Declare an auxiliary array countArray[] of size max(inputArray[])+1 and initialize it with 0.
- Traverse array inputArray[] and map each element of inputArray[] as an index of countArray[] array, i.e., execute countArray[inputArray[i]]++ for 0 <= i < N.
- Calculate the prefix sum at every index of array inputArray[].
- Create an array outputArray[] of size N.
- Traverse array inputArray[] from end and update outputArray[countArray[inputArray[i]] – 1] = inputArray[i]. Also, update countArray[inputArray[i]] = countArray[inputArray[i]]- – .

Advantage of Counting Sort:

- Counting sort generally performs faster than all comparison-based sorting algorithms, such as merge sort and quicksort, if the range of input is of the order of the number of input.
- Counting sort is easy to code
- Counting sort is a stable algorithm.

Disadvantage of Counting Sort:

- Counting sort doesn’t work on decimal values.
- Counting sort is inefficient if the range of values to be sorted is very large.
- Counting sort is not an In-place sorting algorithm, It uses extra space for sorting the array elements.

Applications of Counting Sort:

- It is a commonly used algorithm for the cases where we have limited range items. For example, sort students by grades, sort events by time, days, months, years.

2. PIGEONHOLE SORT: Pigeonhole sorting is a sorting algorithm that is suitable for sorting lists of elements where the number of elements and the number of possible key values are approximately the same. It requires O(n + Range) time where n is the number of elements in the input array and 'Range' is the number of possible values in the array. The Pigeonhole Sort follows the **Brute Force and Bucket-based algorithmic paradigms.**

Pigeonhole Algorithm:

- Find minimum and maximum values in array. Let the minimum and maximum values be 'min' and 'max' respectively. Also find range as 'max-min+1'.
- Set up an array of initially empty "pigeonholes" the same size as of the range.
- Visit each element of the array and then put each element in its pigeonhole. An element arr[i] is put in hole at index arr[i] – min.
- Start the loop all over the pigeonhole array in order and put the elements from non- empty holes back into the original array.

Advantages of Pigeonhole sort:

- It is a non-comparison based sort making it faster in application.
- It is a stable sorting algorithm.
- It performs sorting in linear time.

Disadvantages of Pigeonhole sort:

- It is not easy to know the range of the numbers to sort.
- This number might only work with zero and positive integers.

Applications of Pigeonhole Sort:
Sock picking, Hand shaking and Hair counting.

REFERENCES:
https://www.geeksforgeeks.org/sorting-algorithms/