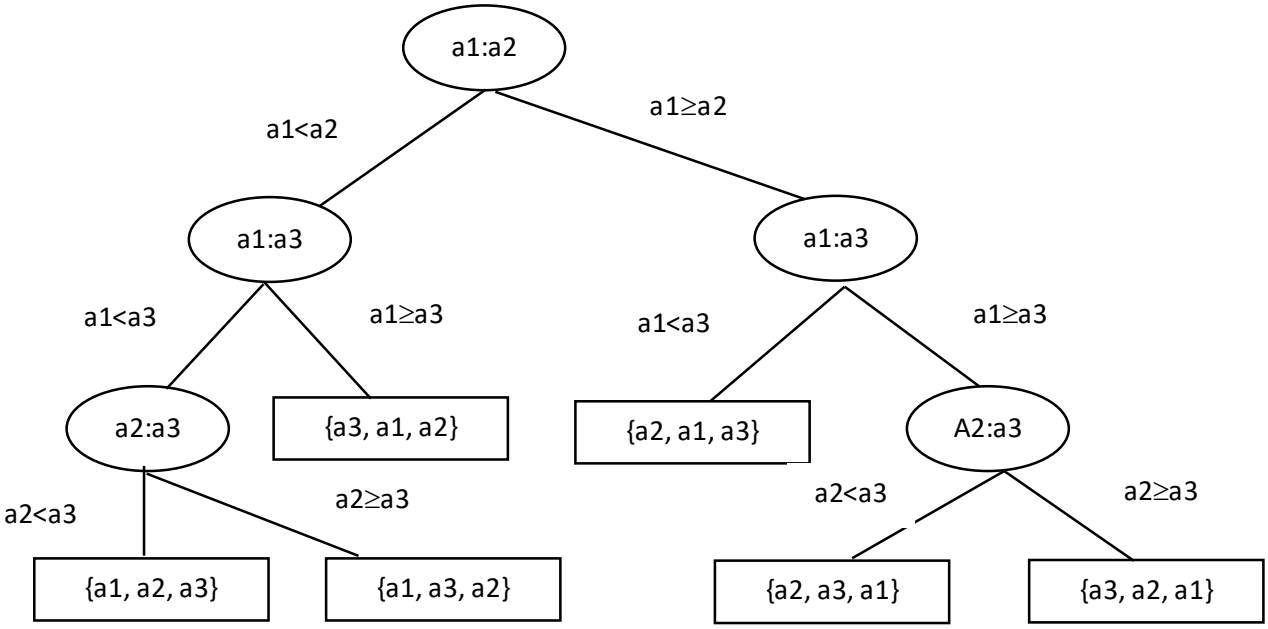
Assignment - 4

Answer to the Question No – 1

The decision tree for deterministic quicksort with an input of size 3 is given below:



Answer to the Question No – 2

Given array – 646, 920, 619, 853, 864, 541, 196, 582, 167, 678, 661

If we start with the most significant digit we will get the following outputs:

Given	Step - 1	Step - 2	Step-3
646	196	619	920
920	167	920	541
619	541	541	661
853	582	646	582
864	646	853	853
541	619	167	864
196	678	661	646
582	661	864	196
167	853	678	167
678	864	582	678
661	920	196	619

So, we get 920, 541, 661, 582, 853, 864, 646, 196, 167, 678, 619 and this array of numbers is not sorted. To overcome this complications, we can partition the numbers into bins by their digits and only sort least significant digits from numbers in the same bin.

Given	Step - 1	Step - 2
646	196	167
920	167	196
619	541	541
853	582	582
864	646	619
541	619	646
196	678	678
582	661	661
167	853	853
678	864	864
661	920	920

In the second step, each bin has 1 element after sorting 2 digits. So, no further sorting is required.

Answer to the Question No – 3

To choose a sorting algorithm with the best worst-case running time possible for an array of n values in the range 0 to 2n, let's compare counting sort and radix sort.

Counting Sort: It counts occurrences of each value in the input array and then the count information is used to place the values in the correct order. The worst-case running time of counting sort is O(n + k), where n is the number of elements in the array and k is the range of possible values. In this case the range is from 0 to 2n, so k = 2n.

Thus, the worst-case running time is O(n + 2n) = O(n).

Radix Sort: It processes the digits if the numbers in multiple passes and typically uses a stable sorting algorithm like counting sort as a subroutine to sort based on each digit. The runtime of radix sort is O(d(n + k)) where d is number of digits, n is the number of elements and k is the range of each digit (e.g each digit of a decimal number can be 0 to 9, so k = 10).

To represent any number we need approximately $\log_2 2n$ number of digits (in binary) for n values within the range 0 to 2n. since, 2n is the maximum value, the number of digits is O(logn).

Radix sort performs O(logn) passes (one for each digit) and each pass can use counting sort to sort the digits. We know counting sort runs in O(n) time. Thus, the total worst-case time complexity of radix sort is O(nlogn).

Comparing the worst-case time complexity of counting and radix sort, we can say the best worst-case running time is O(n). So, it is better to choose counting sort.